한국기록관리학회지
26(2), 131-156, 2026.5
http://dx.doi.org/10.14404/JKSARM.2026.26.2.131
pISSN 1598-1487 eISSN 2671-7247

JKSARM
Journal of Korean Society of Archives and Records Management

웹 기반 행정정보 데이터세트의 XSLT 재현 방안 연구: HTML 구조 유형화를 중심으로

A Study on XSLT-Based Reproduction of Web-Based Administrative Information Datasets: Focusing on HTML Structure Typology

박라미(Lami Park)[1], 양동민(Dongmin Yang)[2]

E-mail: thefirstdrag@jbnu.ac.kr, dmyang@jbnu.ac.kr

[1] 제 1저자 전북대학교 기록관리학과 석사과정
[2] 교신저자 전북대학교 기록관리학과 교수, 문화융복합아카이빙연구소 연구원

논문접수 2026-04-14
최초심사 2026-04-21
게재확정 2026-05-11

ORCID
Lami Park
https://orcid.org/0009-0008-4094-8994
Dongmin Yang
https://orcid.org/0000-0002-4029-9372

▪ 이 논문은 2023년 대한민국 교육부와 한국연구재단의 인문사회분야 중견연구자지원사업의 지원을 받아 수행된 연구임 (NRF-2023S1A5A2A01077759).
▪ 이 논문은 2025년도 전북대학교 연구기반 조성비 지원에 의하여 연구되었음.

초 록

현행 기록관리 체계는 행정정보 데이터세트의 원천 데이터(Raw Data) 보존에는 기술적으로 유효하나, 사용자 인터페이스와 같은 재현 정보를 포함하지 않아 산 당시의 구조와 맥락을 충분히 전달하는 데 한계가 있다. 이에 본 연구는 웹 기반 행정정보시스템에서 생성되는 행정정보 데이터세트의 재현을 위해 HTML 구조를 분석하고, XSLT 기반 재현 방안의 적용 가능성을 실증적으로 검증하였다. 이를 위해 공공기관 행정정보시스템 관련 웹 서비스 10개 사례를 수집하고, 문법 적합성, 구조 복잡도, 화면 복잡도, 스타일/스크립트 의존성, 동적 기능 수준의 다섯 가지 기준에 따라 HTML 구조를 분석하였다. 분석을 통해 재현 대상 HTML은 정적 내재형, 클라이언트-서버 조회형, 다중화면 구성형, 동적 렌더링형의 네 가지 유형으로 구분되었다. 또한 각 유형의 대표 사례를 대상으로 XSLT를 설계·적용하여 재현 가능 범위와 한계를 검증하였다. 검증 결과, 정적 내재형에서 XSLT 재현 방식이 가장 효과적으로 적용되었으며, 외부 실행 환경에 대한 의존도가 높을수록 재현 가능 범위는 축소되고 적용상의 한계가 뚜렷하게 나타났다. 본 연구는 재현 대상 HTML의 유형을 체계적으로 구분하고, XSLT 재현 방안의 적용 가능성과 한계를 실증적으로 제시하였다는 점에서 의의를 가진다.

ABSTRACT

The current records management framework is technically effective in preserving the raw data of administrative information datasets; however, it does not encompass representational information such as user interfaces, which limits its capacity to convey the structure and context of records as they existed at the time of their creation. In response, this study analyzes hypertext markup language (HTML) structures of administrative information datasets generated by web-based administrative information systems and empirically examines the applicability of extensible stylesheet language transformations (XSLT)–based reproduction methods. To achieve this, 10 web services from public institution administrative information systems were selected and analyzed based on five criteria: syntactic validity, structural complexity, visual complexity, style and script dependency, and dynamic functionality level. Through this analysis, HTML structures were classified into four types: static embedded, client–server retrieval, multipage composition, and dynamic rendering. Furthermore, XSLT was designed and applied to representative cases of each type to evaluate the scope and limitations of reproduction. The results indicate that the XSLT-based reproduction approach was most effective for the static embedded type, while reproducibility decreased and limitations became more evident as dependency on external execution environments increased. The significance of the study lies in its systematic classification of reproducible HTML types and its empirical demonstration of the applicability and limitations of XSLT-based reproduction methods.

Keywords: 행정정보 데이터세트, 행정정보 데이터세트 재현, 사용자 인터페이스, HTML, XSLT
Administrative information dataset, administrative information dataset reproduction, user interface, HTML, XSLT

1. 서 론

1.1 연구배경 및 필요성

행정정보 데이터세트는 행정정보시스템에서 전자적 형태로 생산・관리되는 기록으로, 행정업무 수행 과정에서 생산된 정보와 그 맥락을 포함한다. 이때 행정정보시스템은 행정기관이 행정정보를 생산・수집・가공・저장・검색・제공・송신・수신하고 활용할 수 있도록 하드웨어・소프트웨어・데이터베이스를 통합한 시스템을 의미하며, 이를 통해 생성되는 행정정보 데이터세트는 구조화된 테이블 형태의 데이터 집합으로 정의된다.

행정정보 데이터세트는 행정정보시스템의 종료 및 고도화 과정에서 기존 화면과 함께 사용자 인터페이스가 소실될 수 있으므로 기록의 진본성과 맥락을 유지, 표현하기 위해서는 생산 당시의 사용자 인터페이스를 함께 재현하는 것이 필수적이다(변우영, 임진희, 2022). 따라서 행정정보 데이터세트의 재현은 단순히 데이터를 다시 제시하는 것에 그치지 않고, 기록의 내용은 물론 구조와 맥락을 포함한 '외형'이 사용자가 해석 가능한 형태로 제공될 때 비로소 온전한 재현이 성립한다(왕호성, 설문원, 2017). 한편, 현행 기록관리 체계에서는 SIARD_KR과 같은 데이터베이스 기반 이관 도구를 활용하여 행정정보 데이터세트를 XML 형태로 보존하고 있다. 해당 방식은 데이터세트 자체의 보존에는 효과적이지만 화면 구성 정보나 사용자 인터페이스와 같은 재현 정보는 포함하지 않는다. 이로 인해 실제 행정업무 화면에서 제공되던 표 구조, 시각적 배열, 사용자 인터페이스 요소 등은 보존되지 않으며 결과적으로 기록의 'Look & Feel'을 유지하는 데 한계가 발생한다. 그 결과, 행정정보 시스템의 교체 및 고도화 과정에서 기존 화면과 사용자 인터페이스가 소실되고 데이터세트만 이관된 형태로 보존되는 경우가 발생한다. 이러한 방식은 기록의 내용을 유지하는 데에는 유효하나, 생산 당시의 구조와 맥락이 충분히 전달되지 않아 재현에 한계가 따른다. 이로 인해 사용자가 데이터세트의 의미를 정확히 해석하기 어려워지며 나아가 기록의 증거적 가치 저하로도 이어질 수 있다(왕호성, 설문원, 2017). 이와 같은 한계를 보완하기 위해 재현 기능을 포함한 정보시스템을 별도로 구축하는 방식(에뮬레이션)이나 화면 캡처 기반 보존 방식이 제시되어 왔다. 그러나 에뮬레이션은 운영 환경 전체를 포함하는 고비용・고난도의 인프라를 요구하며(Acker, 2021), 화면 캡처 방식은 전자기록의 구조적 특성을 충분히 반영하지 못한다는 한계를 가진다(왕호성, 설문원, 2017).

이에 왕호성과 설문원(2017), 변우영과 임진희(2022)는 행정정보시스템이 제공하는 사용자 인터페이스 화면 자체를 서식 정보로 간주하여 함께 보존하는 방안을 제시하였다. 사용자 인터페이스는 다양한 매체와 기술 환경에 따라 여러 형태로 제공될 수 있다. 예를 들어 웹페이지, 데스크톱 응용프로그램, 모바일 앱, 전자문서 기반 서식, 키오스크형 단말 인터페이스 등이 이에 해당한다. 이 중 HTML 기반의 웹페이지는 가장 보편적인 구현 방식 가운데 하나로, 사용자의 요청에 따라 데이터베이스에서 데이터세트를 조회하고 그 결과를 화면에 구현하는 구조를 가진다. 이러한 점에서 HTML 웹페이지는 단순한 표시 수단을 넘어, 데이터세트의 구조와 표현 방식이 결합된 서식 정보로 이해할 수 있다. 따라서 데이터세트가 표출되는 HTML 태그와 행정정보 데이터세트의 컬럼 간 대응관계를 설정할 수 있다면 단순 화면 캡처보다 실제 화면의 구조와 표현을 보다 충실하게 재현할 수 있을 것이다. 따라서 행정정보 데이터세트의 보존은 데이터뿐만 아니라 HTML과 같은 사용자 인터페이스 기반 재현 요소를 함께 고려해야 하며, 이에 XML과 연계될 수 있는 XSLT 재현 방안이 유력한 대안으로 검토된다.

그러나 XSLT 재현 방안의 필요성은 선행연구를 통해 제기되어 왔음에도 불구하고, 실제 행정정보시스템의 다양한 HTML 구조를 대상으로 한 실증적 적용 연구는 아직 미비한 상황이다. 특히 행정정보 시스템의 사용자 인터페이스는 구현 방식과 기술 환경에 따라 HTML 구조가 상이하게 나타나므로 재현 대상 HTML의 구조적 특성을 유형화하고 이에 따른 XSLT 재현 방안의 적용 가능성을 실증적으로 검토하는 연구가 요구된다. 이에 본 연구에서는 공공기관 행정정보시스템 관련 사용자 인터페이스 사례를 수집하여 HTML 구조를 분석하고, 유형별로 XSLT를 설계・적용하여 XSLT 재현 방안의 적용 가능성을 검증하고자 한다.

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

1.2 선행연구

선행연구는 행정정보 데이터세트 보존에 관한 연구, 전자기록물의 재현에 관한 연구, HTML 기반 XML/XSLT 변환에 관한 연구로 세 가지로 구분된다.

먼저 행정정보 데이터세트 보존에 관한 연구는 주로 데이터베이스 기반 보존 방식과 그 한계를 분석하는 데 초점을 두고 이루어져 왔다. 김수영(2023)은 DAMS와 SIARD_KR을 활용한 실제 이관 사례를 중심으로 데이터세트 이관 과정에서 발생하는 문제를 분석하고 다양한 이관 방식을 제안하였다. 특히 SIARD_KR을 실데이터에 적용하여 나타나는 오류 유형을 정리하고, DB 덤프 방식, 오픈포맷 방식, 비정형 데이터베이스 이관 등 대안적 접근을 제시하였다. 변우영과 임진희(2022)는 SIARD_KR이 단순한 데이터 추출 도구에 그치는지에 관한 여부를 문제로 제기하며, 행정정보 데이터세트의 의미 단위를 반영할 수 있는지에 주목하였다. 이 연구는 데이터 전체가 아닌 '의미 있는 정보'의 식별 가능성과 시스템 분리 이후에도 정보의 의미가 유지되는지를 검토하였다. 또한 윤성호 외(2021)는 SIARD 포맷의 보존 가능 범위를 검증하기 위해 실험적 검증을 수행하고, 특정 데이터 타입과 구조 요소가 보존되지 않는 한계를 확인하였다. 한희정 외(2020)는 데이터세트의 특성을 반영한 보존포맷 선정 기준과 평가체계의 필요성을 제기하며, 실증적 검증을 통한 보존 전략 수립의 중요성을 강조하였다.

전자기록물의 재현에 관한 연구에서는 데이터뿐 아니라 사용자 인터페이스의 중요성이 점차 강조되고 있다. 왕호성과 설문원(2017)은 데이터세트를 전자기록으로 관리하기 위해 재현성이 필수적임을 강조하고, 기록의 진본성은 내용, 구조, 맥락뿐만 아니라 외형과 기능까지 함께 보존될 때 확보된다고 보았다. 이러한 논의는 데이터세트 이외에도 사용자 인터페이스 역시 보존 대상에 포함되어야 함을 시사한다. 이에 양동민 외(2023)는 필수보존속성 관점에서 사용자 인터페이스의 보존 필요성을 강조하고, 이를 XSLT로 재현하는 방법론을 제시하였다.

마지막으로 HTML 기반 XML/XSLT 변환에 관한 연구에서는 다음과 같은 논의가 이루어졌다. 오금용과 황인준(2002)은 HTML 문서의 반복적인 구조 패턴을 분석하여 XML로 자동 변환하는 방법을 제안하였으며, 문서 구조를 기반으로 변환 경로를 인식하는 방식을 구현하였다. 또한 유상원 외(2003)는 XML이 가진 스키마 표현 능력과 구조 정보를 이용하여 사용자 정보를 기술하는 방법을 제시하였으며, 이러한 사용자 정보는 XML 질의에 반영되어 XML로 이루어진 문서를 통해 전달된다.

선행연구를 종합하면, 행정정보 데이터세트 연구는 데이터세트 이관 및 보존 포맷에 집중되어 있으며, 재현 연구는 사용자 인터페이스의 중요성을 제기하는 수준에 머물러 있다. 또한 HTML 변환 연구에 있어 XML을 활용한 기술적 방법론이 일부 제시되었으나, 사용자 인터페이스를 보존하기 위해 이를 재현하는 XSLT 설계에 관한 연구는 미비한 실정이다. 이에 따라 기존 연구에서는 다음과 같은 한계가 확인된다. 첫째, 데이터 중심 보존 방식으로 인해 사용자 인터페이스에 관한 재현 요소가 충분히 고려되지 않았다. 둘째, 사용자 인터페이스 기반 재현 필요성이 제기되었으나 실제 적용 가능한 기술적 절차와 방법이 구체적으로 제시되지 않았다. 셋째, HTML 구조를 유지한 상태에서 사용자 인터페이스를 재현하기 위한 XSLT 설계 및 적용 연구가 부족하다.

본 연구는 이러한 한계를 보완하기 위해 실제 행정정보 데이터세트 재현 대상 사례를 수집하여 각 HTML 구조를 직접 분석하고 재현 가능성을 판단하고자 하였다. 이를 위해 HTML 유형을 도출하고, 각 유형별로 원본 HTML의 구조와 표현을 가능한 한 그대로 반영하는 XSLT를 설계·적용하였다. 더하여 각 유형별로 실제 사용자 인터페이스가 어느 수준까지 재현되는지를 검증함으로써 HTML을 변형하지 않고 그대로 재현하기 위한 XSLT 재현 방안의 적용 가능성과 한계를 실증적으로 제시하고자 한다.

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

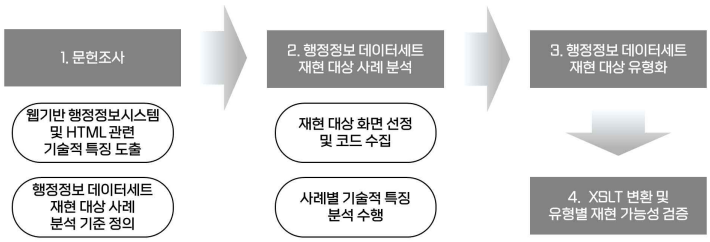
1.3 연구방법

본 연구에서는 행정정보 데이터세트 재현 대상의 HTML을 분석하고, HTML 기반 XSLT 재현 방안의 적용 가능성을 검토하기 위해 <그림 1>과 같은 연구방법을 적용하였다.

먼저 HTML, XSL/XSLT, 행정정보 데이터세트의 XSLT 재현 방안과 관련된 선행연구 및 기술 문헌을 검토하였다. 이를 통해 HTML의 기술적 특징을 중심으로 XSLT 재현에 영향을 미치는 주요 요소를 도출하고, 재현 대상 HTML의 분석 기준을 설정하였다.

다음으로 공공기관에서 제공하는 행정정보시스템 관련 웹 서비스 중 다양한 사용자 인터페이스 및 웹 구현 방식을 포괄할 수 있도록 10개 사례를 선정하여 각 사례의 HTML 코드 및 재현에 필요한 외부자원을 수집하였다. 사례 선정 과정에서는 특정 서비스 유형에 편중되지 않도록 하였으며, 데이터 제공 방식과 화면 구성 방식에서 상이한 특성을 보이는 사례를 포함하여 분석 대상의 다양성을 확보하였다. 각 사례에 대해서는 웹 브라우저를 통해 실제 화면을 확인한 후, 선행연구 검토를 통해 도출된 분석 기준에 따라 수집한 HTML 코드를 관찰, 분석하였다. 이후 사례 분석 결과를 바탕으로 공통적인 구조적 패턴을 중심으로 유형을 도출하였다. 또한 각 유형의 대표 사례를 선정하여 XSLT를 설계・적용하였다. 이때 적용 범위는 데이터 조회 및 결과 표출 화면으로 한정하고, 팝업 문구와 같은 비핵심 요소는 제외하였다. 이는 재현에 핵심적인 데이터 구조와 표현 요소에 분석을 집중하고, 비핵심 요소로 인한 영향 개입을 최소화하기 위함이다.

마지막으로 XSLT를 다시 HTML로 변환하여 재현 가능 여부와 그 정도를 파악하고, 구조 및 표현 측면에서의 재현 가능 범위를 분석하였다. 이를 통해 HTML 기반 XSLT 재현 방안의 적용 가능성과 한계를 도출하였다.



〈그림 1〉 행정정보 데이터세트 재현을 위한 HTML 유형화 및 XSLT 재현 방안 적용을 위한 연구방법

2. 이론적 배경

2.1 HTML

W3C에서 발표한 웹 표준 권고안에 따르면, 웹 문서는 구조・표현・기능이 분리된 형태로 구성되며 이는 각각 HTML(Hypertext Markup Language), CSS(Cascading Style Sheets), JavaScript에 의해 구현된다(W3C, 2014). HTML은 웹 문서의 구조를 정의하고, CSS는 레이아웃과 색상, 글꼴 등 시각적 표현을 담당하며, JavaScript는 사용자 인터랙션과 같은 동적 기능을 수행한다. 이처럼 세 요소가 분리된 구조는 서로 다른 브라우저 환경에서도 공통된 방식으로 웹 문서를 제공할 수 있게 하며, 별도의 브라우저별 웹사이트를 따로 제작하지 않아도 동일한 내용을 표시할 수 있도록 한다(이채혁, 2012).

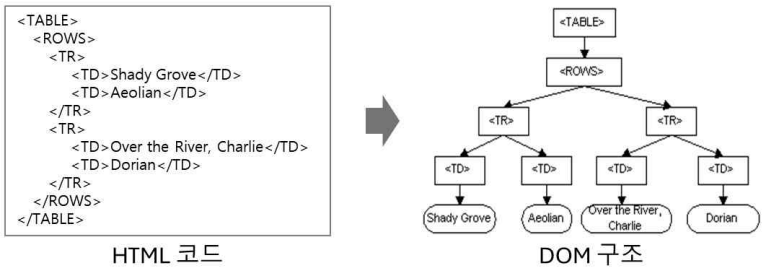
HTML은 웹상에서 과학 관련 문서를 공유하고 컴퓨터가 이해할 수 있는 구조화된 문서를 표현하기 위해 개발

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

된 마크업 언어로, <그림 2>와 같이 태그를 이용하여 문서의 구조를 기술하고 그 안에 내용을 배치하는 방식으로 구성된다. W3C(World Wide Web Consortium)에 등록된 최초의 공식 HTML 표준은 3.2 버전이며, 이후 HTML 4를 거쳐 현재는 HTML 5.3까지 발전하였다(WHATWG, 2026).

〈그림 2〉 HTML 코드 및 실행 화면 예시

HTML은 브라우저에 의해 해석되는 과정에서 <그림 3>과 같이 DOM(Document Object Model) 구조로 변환된다(Robie, 1998). DOM은 HTML의 구성 요소를 객체로 표현하여 트리 구조로 형상화한 것으로, document, form, window 등 문서의 각 속성을 객체 단위로 다룰 수 있게 한다. 이를 통해 HTML 요소에 대한 접근과 이벤트 처리가 가능해지며 현재 대부분의 브라우저는 표준 W3C DOM을 지원한다. 따라서 HTML이 구조적으로 적절히 작성되어 있을 경우, DOM을 통해 트리 형태로 해석하고 이를 기반으로 문서의 구조를 파악할 수 있다. 이러한 DOM의 트리 구조는 특정 요소의 위치와 계층 관계를 체계적으로 탐색할 수 있게 하므로 웹 문서 내에서 표·목록·제목 등 반복적으로 나타나는 데이터 구조를 자동으로 식별하고 추출하는 데 활용될 수 있다(이채혁, 2012). 이는 행정정보 데이터세트와 같이 정형화된 정보가 웹 화면으로 제공될 때 해당 화면의 구조를 분석하고 데이터를 체계적으로 표현하는 기술적 기반으로 기능한다.



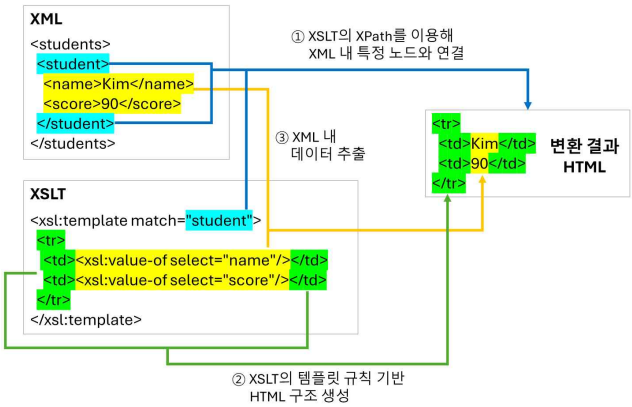
〈그림 3〉 HTML 코드 기반 DOM 구조 변환 예시(W3C, 1999)

2.2 XSL/XSLT

XSL(eXtensible Stylesheet Language)은 XML 표준을 구성하는 세부표준 중 하나로, XML 데이터를 다른 파일 포맷으로 변환할 때 사용되며 XSLT(XSL Transformation), XPath(XML Path Language), XSL-FO(XSL Formatting Object)의 3개 파트로 구성된다. 이 중 XSLT는 구조화된 XML 문서를 다른 형태의 문서로 변환하기 위한 언어로, 변환 규칙을 정의한 스타일시트 형태로 표현되며 HTML과 같은 표현 중심의 문서를 포함한 다양한 구조의 문서를 생성·변환하는 기능을 수행한다(W3C, 2017). 변환 과정에서 XML 문서는 트리 구조로 해석되며, 문서를 구성하는 노드(Node) 단위를 기준으로 처리된다. 이때 노드란 XML 트리 구조에서 데이터의 상하위 계층을 나타내는 항목으로, 요소(element), 속성(attribute), 텍스트(text) 등을 포함하는 개념이다.

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

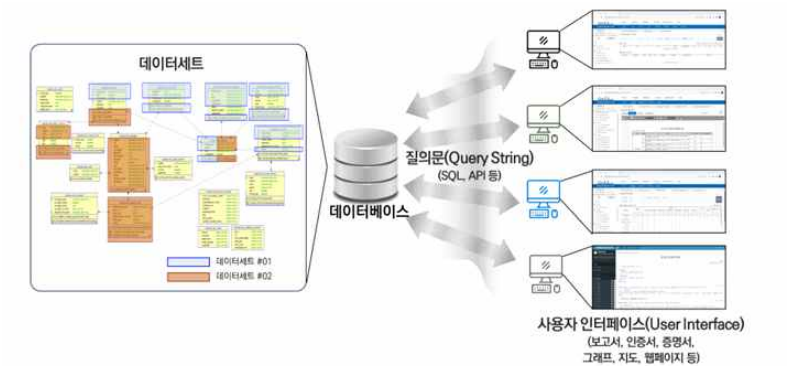
XSLT를 XML과 결합하여 HTML 문서로 변환하는 과정은 <그림 4>와 같이 XML 문서의 구조를 기반으로 특정 요소를 선택하고 이에 대응되는 출력 구조를 생성하는 방식으로 이루어진다. 먼저, XSLT는 XPath를 이용하여 XML 문서 내에서 특정 노드를 선택한다. XPath는 XML 구조를 따라 원하는 요소를 지정하는 경로 표현 방식으로 '<xsl:template match="student">'와 같은 명령을 통해 "student" 노드를 변환 대상으로 설정할 수 있다. 다음으로 선택된 노드에 대해 템플릿 규칙(template rule)이 적용된다. 템플릿 규칙은 특정 노드에 대응되는 출력 구조를 정의하는 것으로, <tr>, <td> 등의 HTML 요소를 생성하도록 설정할 수 있다. 이 과정에서 '<xsl:value-of select=...>'를 이용하여 노드 내부의 데이터를 추출하고 이를 HTML 요소의 내용으로 삽입한다. 이와 같이 XSLT 변환은 노드의 선택과 템플릿 규칙 적용을 통해 XML 데이터를 재구성하여 HTML 문서를 생성하는 과정으로 이루어진다.



〈그림 4〉 XML/XSLT 기반 HTML 변환 과정

2.3 행정정보 데이터세트의 XSL/XSLT 재현 방안

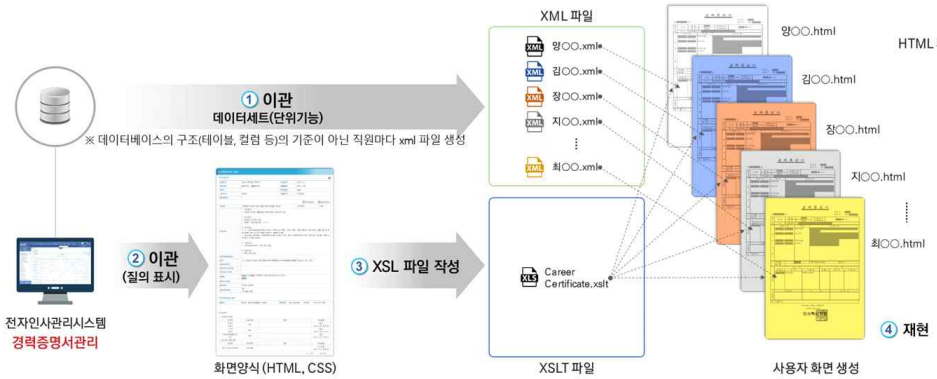
행정정보시스템은 행정 지원을 목적으로 하는 다양한 업무 기능으로 구성되며, 다수의 사용자가 실시간으로 동시에 접속하여 데이터를 공유하며 처리하는 구조를 가진다. 이러한 환경에서 데이터베이스는 정제된 데이터세트를 저장하고, 사용자 인터페이스는 사용자가 필요로 하는 데이터세트를 데이터베이스로부터 불러와 미리 설계된 화면 구성에 맞게 표시하는 역할을 담당한다(<그림 5> 참조). 사용자 인터페이스는 SQL 질의나 API 호출을 통해 데이터를 전달받아 보고서, 증명서, 그래프, 지도 등 다양한 형태로 출력하며, 이들 결과물은 최종적으로 HTML 기반의 웹 문서 형태로 제공된다(양동민, 2023).



〈그림 5〉 데이터베이스와 사용자 인터페이스(양동민, 2023)

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

따라서 행정정보 데이터세트의 재현은 행정정보시스템에서 제공되는 화면을 기준으로 데이터세트와 화면 구성 요소를 결합하여 원본과 동일한 형태의 결과를 생성하는 과정으로 이해할 수 있다(<그림 6> 참조). 이를 위해서는 데이터의 구조를 유지한 상태로 이관하는 것이 전제되어야 하며, 현행 행정정보 데이터세트 이관 도구인 SIARD_KR에서 생성된 XML 형태의 이관 결과물은 데이터의 계층 구조를 유지한 채 행정정보 데이터세트를 표현할 수 있어 XSLT와의 결합이 가능하다. XSLT는 이러한 XML 데이터와 결합하여 데이터값을 미리 정의된 화면 구성에 맞게 배치하는 변환 역할을 수행한 뒤 기존 사용자 인터페이스에 대응되는 재현 결과 HTML 문서를 생성한다. XSLT는 사전에 정의된 형태로 재사용이 가능하므로, 동일한 화면 구조를 가진 데이터세트에 동일한 규칙을 반복 적용하여 일관된 형태의 재현 결과를 생성할 수 있다.



〈그림 6〉 사용자 인터페이스의 XSLT 재현 방안(양동민, 2023)

행정정보시스템의 사용자 인터페이스는 대부분 HTML로 표현되므로, 수집한 HTML 구조를 분석하여 이를 바탕으로 XSLT를 직접 설계할 수 있다는 점에서 XSLT 재현 방안은 실질적인 적용 가능성을 지닌다. 구체적으로는 수집한 HTML의 화면 구조를 분석하여 XSLT를 작성하고, 이를 XML 데이터와 결합하여 HTML 문서를 생성하는 방식으로 이루어진다. 결과적으로 행정정보 데이터세트의 재현은 기존 사용자 인터페이스(HTML), 데이터(XML), 변환 규칙(XSLT)의 세 요소의 결합을 통해 구현되며, 이는 데이터의 구조를 유지하면서 원래의 화면을 재구성할 수 있는 기술적 기반으로 기능한다.

한편, 이렇게 재구성된 화면은 브라우저의 종류나 버전에 따라 원본 화면과 미세한 시각적 차이가 발생할 수 있다. 이는 CSS나 JavaScript와 같은 소스 코드 자체의 변형이 아니라, 브라우저의 해석 방식 차이에서 비롯된 것이다. 재현 결과물인 HTML 코드에서는 화면 구조나 폰트 등 내부 정보가 DOM 구조의 변경 없이 유지되므로, 기록의 증거적 가치에는 실질적인 영향이 없다고 판단된다. 다만 시각적 차이의 주요 원인 중 하나는 날짜 등 외부 연계 데이터나 부수적 요소가 재현 시점 환경에서 정상적으로 표시되지 않아 화면 일부나 정보가 불완전하게 나타나는 경우이다. 이러한 요소들은 보존 대상 화면의 핵심 재현 범위에 포함되지 않는 경우가 많으므로, 업무 담당자와 기록연구사의 협의를 통해 XSLT 재구성 범위를 결정하고, 그 판단 근거와 처리 절차를 기록으로 남겨야 한다.

3. 행정정보 데이터세트 재현 대상 유형화

3.1 분석 기준 설정

행정정보시스템은 구현 방식과 기술 환경에 따라 HTML 구조가 상이하게 나타나므로, 재현 대상 HTML의

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

특성을 사전에 파악하고 유형화하는 것은 XSLT 재현 방안의 적절한 재현 대상을 선택하고 적용 가능성을 판단하는 데 있어 선행되어야 할 작업이다. 이에 본 연구에서는 사례 분석에 앞서 각 사례 HTML의 기술적 복잡도와 구조적 특성을 파악하기 위해 관련 선행연구와 웹 표준 사양을 검토한 후, 문법 적합성, 구조 복잡도, 화면 복잡도, 스타일/스크립트 의존성, 동적 기능 수준과 같은 다섯 가지 분석 기준을 도출하였다. 각 항목에 대한 세부사항은 후술하도록 한다.

3.1.1 문법 적합성

문법 적합성은 HTML을 오류 없는 DOM 트리 구조로 표현하여 XSLT 변환을 수행하기 위한 기준이다. HTML 문법 오류, 태그 생략, 특수기호 처리 방식 등은 안정적 문서 실행 여부에 직접적인 영향을 미치며, 이는 HTML 기반 XSLT 변환 이전 단계에서 반드시 확인되어야 한다. 이에 본 연구에서는 문법 적합성을 HTML의 적격(Well-formed) 여부를 통해 분석하고자 한다. Well-formed인 HTML은 문법적 오류 없이 작성된 문서로, 웹 브라우저에 의해 DOM 트리 구조로 안정적으로 구성될 수 있으며, 이를 바탕으로 데이터의 삽입과 추출이 가능한 구조를 가진다. Vodnik(2011)은 Well-formed인 HTML의 조건으로 다음과 같은 조건을 정의하였다. 첫째, 모든 엘리먼트들은 시작 태그와 종료 태그를 가져야 한다. 둘째, 모든 태그는 올바르게 중첩되어야 한다. 셋째, 모든 속성값들은 따옴표로 묶여있어야 한다. 넷째, 모든 엘리먼트들이 빈 태그로 사용될 때는 “/>”로 끝나야 한다. 이와 같이 4가지 조건이 충족된 HTML 문서는 태그와 텍스트뿐만 아니라 다른 웹 문서로의 링크, 테이블, 이미지와 같은 멀티미디어 데이터를 포함하는 구조화된 HTML 문서로 해석될 수 있다(오금용, 황인준, 2002). 따라서 문법 적합성은 단순한 실행 오류 점검을 넘어, 재현 대상 HTML의 구조 분석과 재현 가능성을 판단하는 기술적 무결성의 지표로 볼 수 있다. 이에 본 연구에서는 Vodnik(2011)이 제시한 4가지 조건을 기준으로 HTML의 문법 적합성을 평가하며, 해당 조건을 충족하는 경우 문법 적합성이 확보된 것으로 정의하였다.

3.1.2 구조 복잡도

Chamberland-Thibeault와 Hallé(2020)는 HTML 기반 웹사이트의 구조적 특성을 정량화하기 위해 DOM 노드 수, DOM 트리 깊이, 태그 유형 분포를 주요 분석 지표로 설정하고, DOM 구조 추출 과정에서도 태그 종류, 트리 깊이와 같은 정보를 수집하였다. 이는 웹 페이지의 구조를 구성하는 요소를 계량화하여 사이트의 구조적 특성과 복잡도를 체계적으로 파악하기 위한 접근으로, 웹 구조 분석에서 활용되는 대표적인 지표로 볼 수 있다. 따라서 이러한 지표들은 HTML 문서의 구조적 특성을 정량적으로 설명할 수 있는 기준으로 작용하는데, DOM 노드 수와 트리 깊이는 HTML의 구조적 밀도와 계층적 복잡성을 보여주며, 태그 종류는 HTML을 구성하는 표현 방식의 다양성을 나타낸다(Clark, 2025). 또한 링크 구조는 HTML 내 탐색 요소와 상호 연결 구조를 파악하는 데 활용될 수 있다.

따라서 본 연구는 이러한 구조적 특성과 데이터 구조의 복잡도를 종합적으로 고려하여 구조 복잡도를 XSLT 변환과 연관된 분석 기준으로 설정하였다. 구조 복잡도의 세부 기준은 <표 1>과 같다.

〈표 1〉 구조 복잡도 세부 항목

세부 항목	내용
DOM 노드 수	총 element 노드
트리 최대 깊이	최상위 요소에서 가장 깊은 요소까지 단계 수 측정
태그 종류 수	고유 element 유형 수
링크 수 〈a〉	〈a href〉 개수

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

3.1.3 화면 복잡도

화면 복잡도는 사용자 인터페이스 내에서 사용자가 인지하는 시각적 구성의 다양성과 배치 구조를 파악하기 위한 기준으로, 텍스트, 이미지, 입력 요소, 표, 차트, 지도 등 화면을 구성하는 시각 객체의 수와 유형, 그리고 영역 간 배치 방식에 의해 결정된다. 이러한 화면 복잡도는 HTML의 구성 방식과 밀접하게 연결된다. HTML은 DOM 트리 기반의 레이아웃 영역 구조를 통해 화면을 구성하는 주요 영역을 구분할 수 있으며, 이러한 구조는 사용자가 인지하는 복잡도와 밀접한 관련을 가진다(Han et al., 2024). 이는 XSLT 템플릿의 수와 분기 구조 증가로 이어져 변환 난이도를 높이는 요인으로 작용한다. 또한 웹페이지 내 콘텐츠 구성 측면에서도 화면 복잡도를 설명할 수 있다. Michailidou et al.(2008)은 콘텐츠 객체 수와 유형의 다양성이 시각적 복잡도나 렌더링 비용과 밀접하게 관련된다고 보았으며, 시각 객체 수와 레이아웃 범위가 화면 구조의 복잡도와 화면 계산 비용에 영향을 미친다고 설명한다.

본 연구에서는 이러한 논의를 바탕으로 화면 복잡도를 분석하기 위해 '시각 객체 수', '레이아웃 영역 수', '차트 표현 방식', '콘텐츠 유형 구성', '지도/차트 외부 라이브러리 사용 여부'를 세부 기준으로 설정하였다(<표 2> 참조). 특히 '지도/차트 외부 라이브러리 사용 여부' 항목에서 외부 라이브러리 기반 시각화 방식이 사용된 경우에는 정적 HTML만으로 재현 가능한지에 관한 여부를 함께 검토하였다.

〈표 2〉 화면 복잡도 세부 항목

세부 항목	내용
시각 객체 수	이미지(img), 입력 요소(input), 표(table), 그래픽 요소(svg, canvas) 등 시각 구성 요소의 총합
레이아웃 영역 수	id/class 기준 주요 영역(헤더, 내비게이션, 본문, 푸터 등) 분리 개수
차트 표현 방식	JavaScript 기반 시각화 라이브러리 / canvas / SVG / 〈table〉 태그 기반 표현 / 없음
콘텐츠 유형 구성	텍스트, 표, 이미지, 차트, 지도 등 포함된 주요 콘텐츠 유형 기술
지도/차트 외부 라이브러리 사용	외부 시각화 라이브러리 사용 여부(O / X)

3.1.4 스타일/스크립트 의존성

스타일/스크립트 의존성은 HTML에 사용되는 외부자원, 즉 CSS와 JavaScript가 화면 표현과 기능 구현에 어느 정도 관여하는지를 판단하기 위한 기준이다. CSS와 JavaScript는 HTML의 시각적 표현과 기능을 담당하지만, 그 의존도가 높아질수록 외부자원을 제외한 HTML 자체만으로는 원래의 화면을 XSLT로 재현하기 어려워질 수 있다. 이는 HTML 코드만으로 화면을 재현할 수 있는지 여부를 판단하는 중요한 기준이 된다. Goel et al.(2022)은 사용자에게 최종적으로 제공되는 화면 출력값은 초기 HTML이 단순한 골격을 구성하는 수준에 머무르고, 실제 콘텐츠와 구조는 스크립트와 외부 데이터 응답을 통해 동적으로 생성·갱신된다고 지적하였다. 아울러 이러한 환경에서는 HTML과 함께 CSS, JavaScript, JSON 자산을 별도로 수집하더라도 원본과 동일한 화면과 기능을 재현하기 어려운 사례가 발생할 수 있다고 보았다. 더불어 CSS와 JavaScript는 HTML 포함 방식에 따라 재현에 미치는 영향이 다르게 나타난다. 외부 파일 형태로 참조되는 CSS와 JavaScript는 HTML 외부에 존재하므로 별도의 파일 저장과 연결이 필요하며, 누락될 경우 화면 재현이 불완전해질 수 있다. 반면 인라인 CSS 및 JavaScript는 HTML 내부에 포함되어 있으므로, HTML 구조 분석 및 XSLT 변환 과정에서 직접적인 영향을 미치는 요소로 작용한다. 더불어 번들 JavaScript는 다수의 JavaScript를 하나의 파일로 결합하여 내부 기능과 구조가 압축·은닉되어 있는 경우가 많아 동작 분석과 재현을 어렵게 만드는 요인으로 작용한다.

이러한 논의를 종합하면, 외부 CSS 및 JavaScript 의존성이 높을수록 화면 구성과 기능이 HTML 구조 외부로 분산되며, 이는 구조 기반 변환이나 재현을 어렵게 만드는 요인이 된다. 따라서 본 연구에서는 '외부 CSS 파일 수', '외부 JavaScript 파일 수', '번들 JavaScript 사용 여부', '인라인 CSS 특징 기술', '인라인 JavaScript 특징 기술'

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

항목을 중심으로 각 사례의 스타일/스크립트 의존성을 분석하였다(<표 3> 참조).

〈표 3〉 스타일/스크립트 의존성

세부 항목	내용
외부 CSS 파일 수	〈link rel="stylesheet"〉 태그를 통해 로드되는 외부 CSS 파일의 개수
외부 JS 파일 수	〈script src="..."〉 태그를 통해 로드되는 외부 JavaScript 파일의 개수
번들 JS 사용 여부	webpack, chunk.js 등 번들링된 JavaScript 파일 사용 여부(O / X)
인라인 CSS 특징 기술	HTML 문서 내 〈style〉 태그 또는 인라인 style 속성의 특징 서술
인라인 JS 특징 기술	HTML 문서 내 〈script〉 태그에 직접 작성된 스크립트 코드 특징 서술

3.1.5 동적 기능 수준

동적 기능 수준은 사용자 인터페이스와 그에 포함된 기능이 웹서버에서 획득 가능한 HTML 및 외부자원만으로 재현 가능한지, 혹은 외부 애플리케이션이나 프로그램 실행 환경에 의존하는지를 구분하기 위한 기준이다. 해당 항목은 웹 애플리케이션을 구분하는 유형 분류에 근거하여 설정하였다. Daszkewicz(n.d.)는 정적 페이지, 서버 측 동적 페이지, 클라이언트 측 단일 페이지 애플리케이션(SPA)으로 웹 애플리케이션의 유형을 나누었다. 이때 정적 페이지는 웹서버가 미리 생성한 HTML을 그대로 제공하는 유형이며, 서버 측 동적 페이지는 사용자 요청 시 템플릿을 통해 HTML을 생성하는 유형이고, SPA는 클라이언트 측 JavaScript가 DOM을 지속적으로 갱신하는 유형이다. 이러한 유형들은 XSLT로의 재현 가능성과 직접적으로 연결되는데, 정적 페이지는 서버에서 제공된 HTML 코드만으로도 구조와 기능을 대부분 재현할 수 있는 반면, 동적 페이지와 SPA는 사용자 상호작용 및 데이터 연동에 따라 HTML 구조가 변화하고, 클라이언트 실행 환경에 의존하는 정도가 높아 재현이 어려워진다.

따라서 본 연구에서는 이러한 유형 구분을 바탕으로 서버에서 획득한 HTML만으로 재현 가능한 기능과 클라이언트 실행 환경에 의존하는 기능을 구분하고, 이를 XSLT 기반 재현 가능성 분석의 한 축으로 활용하였다. 이를 위해 '이벤트 처리 구조', 'DOM 조작 코드', '데이터 요청 구조', '비HTML 기반 요소'의 존재 여부를 중심으로 각 사례의 동적 기능 수준을 분석하였다(<표 4> 참조).

〈표 4〉 동적 기능 수준 세부 항목

세부 항목	내용
이벤트 처리 구조	사용자 요청(interaction) 처리를 위한 이벤트 핸들러(onclick, addEventListener 등) 서술
DOM 조작 코드	JavaScript 등을 통한 HTML 구조 생성 및 변경 과정 서술
데이터 요청 구조	외부 데이터 요청(fetch, Ajax 등) 발생 과정 서술
비HTML 기반 요소	iframe, canvas 등 HTML 외부 또는 JavaScript 기반 렌더링 요소 서술

3.2 행정정보 데이터세트 재현 대상 사례 수집 및 분석

본 절에서는 행정정보 데이터세트 재현 대상 사례를 수집하고, 앞서 설정한 분석 기준을 바탕으로 각 사례의 기술적 특성을 분석하였다. 이를 위해 웹 기반 행정정보 데이터세트의 다양한 사용자 인터페이스 구현 방식을 포괄할 수 있도록 공공기관에서 운영 중인 웹 서비스 중 기술적 복잡도와 데이터 노출 방식이 상이한 10개의 사례를 선정하였다(<표 5> 참조). 사례 선정의 주요 기준은 다음과 같다. 첫째, 기관 유형 측면에서는 대학, 중앙행정기관, 지자체, 공공기관을 포함한 다양한 기관 유형을 아우를 수 있도록 하였다. 이는 행정정보 데이터세트 재현 적용 범위가 특정 기관 유형에 국한되지 않기 위한 것이다. 둘째, 서비스 목적 측면에서는 통계 데이터

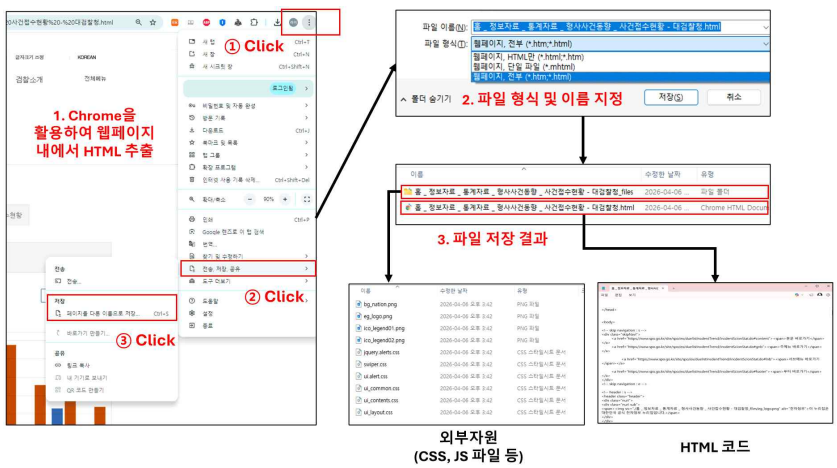
http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

조회, 의사결정 지원 자료 열람, 개인정보 기반 행정서비스, 실시간 모니터링 등 행정정보시스템이 수행하는 다양한 업무 기능을 포괄할 수 있도록 사례를 구성하였다. 셋째, 화면 구성 방식 측면에서는 텍스트 중심 화면, 표 중심 화면, 탭 기반 복합 화면, 지도·차트 복합 화면과 같이 다른 구성 방식이 사례에 포함되도록 하였다.

<그림 7>은 수집 대상 웹페이지의 HTML과 외부자원을 수집하는 절차를 나타낸 것이다. 웹 브라우저의 개발자 도구를 활용하여 페이지 소스를 저장하면 HTML 파일과 함께 CSS, JavaScript 등 관련 외부자원이 동일한 경로에 저장된다. 이를 통해 각 사례의 웹페이지를 구성하는 HTML과 외부자원을 함께 확보하였으며, 이후 사전에 설정된 다섯 가지 분석 기준에 따라 해당 자료를 관찰·분석하였다.

〈표 5〉 행정정보 데이터세트 재현 대상 사례

순번	재현 대상 사례명	기관 유형	서비스 목적	화면 구성 방식
1	J대학 학사관리시스템 인사관리카드	대학	개인정보 기반 행정서비스	탭 기반 복합 화면
2	국회 회의록 조회 화면 (https://record.assembly.go.kr/assembly/)	정부 (국회)	의사결정 지원 자료 열람	텍스트 중심 화면
3	국가통계포털(KOSIS) 데이터 조회 화면 (https://kosis.kr/index/index.do)	정부 (국가데이터처)	통계 데이터 조회	표 중심 화면
4	지표누리(e-나라지표) 조회 화면 (https://www.index.go.kr/enara)	정부 (국가데이터처)	통계 데이터 조회	지도·차트 복합 화면
5	검찰통계시스템 통계자료 조회 화면 (https://www.spo.go.kr/site/spo/crimeAnalysis.do)	정부 (법무부)	통계 데이터 조회	지도·차트 복합 화면
6	서울시 실시간 도시데이터 (https://data.seoul.go.kr/SeoulRtd/)	지자체 (서울시)	실시간 모니터링	지도·차트 복합 화면
7	ICT데이터허브 미디어통계포털 조회 화면 (https://stat.kisdi.re.kr/main.html)	공공기관 (정보통신정책연구원)	통계 데이터 조회	표 중심 화면
8	한국은행 경제통계시스템 조회 화면 (https://ecos.bok.or.kr)	중앙은행	통계 데이터 조회	표 중심 화면
9	국세청 홈택스 소득 세액공제 자료 조회 화면 (https://hometax.go.kr/)	정부 (국세청)	개인정보 기반 행정서비스	탭 기반 복합 화면
10	기상자료개방포털 조회 화면 (https://data.kma.go.kr/cmmn/main.do)	정부 (기상청)	통계 데이터 조회	지도·차트 복합 화면



〈그림 7〉 행정정보 데이터세트 재현 대상의 HTML 코드 및 외부자원 수집 방법

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

분석 결과, ‘문법 적합성’ 항목에서는 모든 사례가 Well-formed인 HTML로 나타났다. ‘구조 복잡도’ 항목에서는 DOM 노드 수가 최소 150개에서 최대 7,756개로, 트리 최대 깊이는 6단계에서 35단계까지, 고유 태그 종류는 14종에서 50종까지 사례별로 넓은 분포를 보였다. 이처럼 ‘구조 복잡도’ 수치는 사례마다 상이하게 나타나 XSLT 변환 과정에서 문서 작성의 난이도를 가늠하는 데에는 유의미한 지표로 작용하였다. 그러나 수치의 분포가 유형 간 경계와 일치하지 않아, ‘구조 복잡도’ 항목만으로는 HTML을 구분하는 기준으로서 사례 간 구조적 차이를 명확히 드러내기에는 한계가 있었다. 이에 본 연구에서는 유형 간 구조적 차이를 직접적으로 반영할 수 있는 ‘화면 복잡도’, ‘스타일・스크립트 의존성’, ‘동적 기능 수준’ 항목을 중심으로 분석 결과를 재구성하였으며, 세부 기준으로 분류되지 않는 특이사항은 ‘기타사항’으로 별도 정리하였다. 이를 바탕으로 사례 간 공통점과 차이점을 함께 검토하여 공통적인 구조적 패턴을 보이는 사례들을 묶어 유형을 도출하고, 그 결과를 <표 6>과 같이 정리하였다. 상세 분석 결과는 <별첨 1>에 제시하였다.

〈표 6〉 행정정보 데이터세트 재현 대상 HTML 분석 결과의 공통 특성 도출 및 해당 사례 정리

사례	분석 기준	분석 결과
J대학 학사관리시스템	화면 복잡도	- vframe・hframe 분할 구조 중첩
	스타일・스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 스타일이 모두 Nexacro 런타임 산출값
	동적 기능 수준	- Nexacro, iframe 포함 - jQuery 비동기 데이터 요청 구조
국회 회의록	화면 복잡도	- 차트 표현 방식: canvas
	스타일・스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 레이아웃 중심
	동적 기능 수준	- 이미지 데이터(TIFF) 비동기 요청 - 서버 측에서 완성된 HTML을 제공
국가통계포털(KOSIS)	화면 복잡도	- 대부분 시각적으로 렌더링되지 않는 파라미터 필드
	스타일・스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- form submit, iframe을 활용한 데이터 요청 구조 - 비HTML 기반 요소에 iframe 포함
e-나라지표	화면 복잡도	- 차트・표・텍스트・UI・링크 복합 구성
	스타일・스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- 차트 영역에서 Ajax, iframe src 변경으로 서버 데이터 반복 요청 - jQuery 등을 이용해 DOM 조작
검찰통계시스템	화면 복잡도	- 차트 표현 방식: SVG
	스타일・스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- 이미지 데이터(TIFF) 비동기 요청 - 차트 데이터가 JS 코드 내 인라인으로 존재 - 서버 측에서 완성된 HTML을 제공
서울시 실시간 도시데이터	화면 복잡도	- 지도・차트・표・이미지・UI・텍스트 복합 - 차트 표현 방식: canvas
	스타일・스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- 외부 JS 모듈과 지도 타일 호출로 실시간 데이터 갱신 - jQuery 등을 이용해 DOM 조작

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

사례	분석 기준	분석 결과
미디어통계포털	화면 복잡도	- 차트 표현 방식: JavaScript 기반(iframe)
	스타일·스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- Ajax, iframe, formsubmit로 데이터 요청 - jQuery 등을 이용해 DOM 조작
한국은행 경제통계시스템	화면 복잡도	- 목록·그리드·버튼 구성 - 차트 표현 방식: 〈table〉
	스타일·스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- webpack chunk 분할 로딩 - realgrid·treegrid·realpivot·SPA 포함
국세청 홈택스	화면 복잡도	- 텍스트·표·UI·링크·탭 복합 - 탭 컨트롤 내부에 iframe 3개가 별도 기능 담당
	스타일·스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - WebSquare 제어에 인라인 스타일 병행
	동적 기능 수준	- WebSquare, iframe으로 데이터 요청 - jQuery 등을 이용해 DOM 조작
기상자료개방포털	화면 복잡도	- 통계표·차트·조회폼·텍스트·목록 구성 - 차트 표현 방식: JavaScript 기반(Echarts)
	스타일·스크립트 의존성	- 외부 CSS, 외부 JS, 번들 JS 사용 - 인라인 CSS는 보조적 수준(글꼴 등)
	동적 기능 수준	- 차트 영역에서 Ajax, Bcharts로 데이터 요청 - jQuery 등을 이용해 DOM 조작

⇓

공통 특성	해당 사례
(유형 1) - 외부 프로그램 및 애플리케이션 사용(Nexacro, WebSquare, SPA) - HTML에 인터페이스 및 실시간 데이터가 존재하지 않음	J대학 학사관리시스템, 국세청 홈택스, 한국은행 경제통계시스템
(유형 2) - 이미지 데이터(TIFF) 비동기 요청 - 서버 측에서 완성된 HTML을 제공	국회회의록, 검찰통계시스템
(유형 3) - 차트 등 일부 영역에서 서버 데이터 요청(Ajax, 외부 JS 모듈, iframe 등 사용) - 외부 CSS, 외부 JS, 번들 JS 사용 - jQuery 등을 이용해 DOM 조작 - 인라인 CSS는 보조적 수준으로 사용됨(국세청 홈택스 제외)	e-나라지표, 기상자료개방포털, 미디어통계포털, 서울시 실시간 도시데이터
(유형 4) - 대부분 시각적으로 렌더링되지 않는 파라미터 필드 - 데이터를 표현하는 대부분의 영역이 form submit, iframe을 활용한 데이터 요청 구조	국가통계포털(KOSIS)

먼저 <유형 1>은 외부 프로그램 또는 애플리케이션 기반으로 화면이 구성되는 사례들로, ‘J대학 학사관리시스템’, ‘국세청 홈택스’, ‘한국은행 경제통계시스템’이 이에 해당한다. 이들 사례는 공통적으로 ‘Nexacro’, ‘WebSquare’, SPA와 같은 외부 프로그램 실행 환경 및 구조에 의존하여 화면이 구성되며, HTML 문서 자체에는 실제 XSLT 변환에 사용 가능한 인터페이스 정보와 데이터가 충분히 포함되지 않는다. 예를 들어 ‘J대학 학사관리시스템’은

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

‘vframe’과 ‘hframe’ 요소가 중첩된 구조와 ‘Nexacro’ 기반 UI 구성을 가지고 실제 화면은 프로그램 실행 결과로 완성되는 특징을 가진다. ‘국세청 홈택스’는 ‘WebSquare’ 기반 웹 컴포넌트와 복수의 ‘iframe’ 요소에 의해 기능이 분리된 구조이며, 화면 구성 요소가 HTML 외부의 실행 환경에 의해 통제된다. ‘한국은행 경제통계시스템’은 SPA 구조와 다양한 JavaScript 컴포넌트를 통해 화면이 동적으로 구성되며, 초기 HTML만으로는 전체 인터페이스를 파악하기 어렵다. 이와 같이 <유형 1>은 HTML이 단순한 컨테이너 역할에 머물고, 실제 화면 구성과 데이터 처리가 외부 애플리케이션 실행 환경에서 이루어진다는 점이 특징적이다.

다음으로 <유형 2>는 서버 측에서 완성된 HTML이 제공되는 사례로, ‘국회 회의록’과 ‘검찰통계시스템’이 이에 해당한다. ‘국회 회의록’과 ‘검찰통계시스템’은 모두 서버에서 생성된 HTML이 사용자 인터페이스의 구조와 데이터를 직접 포함하고 있다는 점에서 공통적인 특징을 보인다. 예를 들어 ‘국회 회의록’의 전체 화면 구조는 서버에서 완성된 HTML로 제공된다. 또한 차트 외의 텍스트 및 표 구조 역시 HTML 내에서 직접 확인이 가능하기 때문에 화면 구성 요소가 정적으로 완결된 형태를 보인다. ‘검찰통계시스템’ 역시 ‘SVG’ 기반 차트를 사용하며, 차트 데이터가 JavaScript 내부에 포함되어 HTML 단계에서 이미 화면 구조와 데이터가 결합된 형태를 보인다. 더불어 표 및 통계 데이터 역시 HTML 구조 내에 포함되어 있어 별도의 동적 처리 없이도 주요 정보 확인이 가능하다. 이러한 사례들은 HTML 자체만으로도 인터페이스의 주요 구성을 확인할 수 있는 정적 구조를 가진다는 점에서 동일한 특성을 가진다.

<유형 3>은 ‘e-나라지표’, ‘서울시 실시간 도시데이터’, ‘미디어통계포털’, ‘기상자료개방포털’과 같이 일부 영역에서만 별도의 데이터 요청이 이루어지는 사례들로 구성된다. 이들 사례를 살펴보면, 기본적인 화면 구조는 HTML에서 확인 가능하지만 차트나 특정 결과 영역에서는 추가적인 데이터 요청이 발생한다. ‘e-나라지표’는 차트 영역에서 ‘Ajax’ 라이브러리와 ‘iframe’ 요소을 통해 서버로부터 데이터를 반복 요청하며, 초기 HTML에는 기본 인터페이스 정보와 정적인 데이터 정보만이 포함되고 실제 차트 데이터는 HTML 로딩 후 서버 호출을 통해 채워지는 구조를 보인다. ‘서울시 실시간 도시데이터’는 지도 및 차트 영역에서 외부 모듈을 통해 실시간 데이터가 갱신되며, HTML 문서만으로는 최신 데이터 상태를 반영하기 어렵다. 또한 ‘미디어통계포털’과 ‘기상자료개방포털’ 역시 차트 및 일부 데이터 영역에서 ‘Ajax’ 라이브러리 기반 데이터 요청 구조를 보이며, 특정 영역에 한정된 동적 데이터 갱신이 이루어진다. 이 경우 초기 화면과 실제 데이터 표현 간 시점 차이가 발생하는 특징을 가진다. 이처럼 <유형 3>은 HTML 기반 정적 구조와 동적인 서버 요청 영역이 부분적으로 결합된 혼합형 구조로 이해할 수 있다.

마지막으로 <유형 4>는 화면에 직접 렌더링되지 않는 파라미터 필드와 요청 중심 구조가 두드러지는 사례로, ‘국가통계포털(KOSIS)’이 이에 해당한다. 이 유형은 사용자 인터페이스 자체보다 데이터 요청 구조가 중심이 된다는 점에서 다른 유형과 구별된다. 즉, 화면에 표시되지 않는 파라미터 값과 요청 흐름이 핵심을 이루며, 실제 데이터 출력 역시 이러한 구조를 통해 이루어진다. ‘국가통계포털(KOSIS)’은 ‘form submit’ 라이브러리와 ‘iframe’ 요소를 활용한 데이터 요청 구조를 기반으로 데이터를 조회하며, 다수의 파라미터 필드를 통해 결과가 생성되는 특징을 보인다. 또한 사용자 입력 조건에 따라 서버 요청이 반복적으로 발생하고, 그 결과가 특정 영역에 반영되는 구조를 가진다. 이 과정에서 HTML 문서는 요청을 전달하는 인터페이스 역할에 가까우며, 실제 데이터 생성은 별도의 처리 과정에 의존한다. 이러한 구조는 시각적 인터페이스보다 서버 요청 메커니즘이 중심이 되는 유형으로 이해할 수 있다.

종합하면, <유형 1>은 외부 실행 환경 의존성이 매우 높아 HTML만으로는 인터페이스와 데이터를 확인하기 어려운 구조이고, <유형 2>는 서버 측에서 완성된 HTML이 제공되어 비교적 직접적인 재현 가능성이 높은 구조이다. 또한 <유형 3>은 기본 화면은 HTML로 확인 가능하지만 차트나 일부 결과 영역이 별도 서버 요청에 의존하는 구조이며, <유형 4>는 화면 자체보다 파라미터와 서버 요청 흐름이 중심이 되는 구조로 나타났다. 즉, 각 사례는 단순히 화면 요소의 수나 복잡성만으로 구분되는 것이 아니라 어디까지가 HTML에 포함되어 있고, 어떤 부분이 외부자원과 동적 요청에 의존하는지에 따라 서로 다른 재현 조건을 형성한다는 점에서 유형화될 수 있었다.

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

3.3 행정정보 데이터세트 재현 대상 유형화

앞선 3.2장의 사례 분석 결과를 토대로 행정정보 데이터세트의 재현 대상 HTML은 <표 7>과 같이 네 가지 유형으로 구분된다.

〈표 7〉 행정정보 데이터세트 재현 대상 HTML 유형

유형		내용	사례
(유형 2) 정적 내재형		• HTML 문서 내에 모든 데이터와 인터페이스 정보가 포함되어 있음 • HTML 로드 시점에 이미 완결된 정적 상태이며, 서버 및 애플리케이션 요청 없이 전체 내용 확인 가능	• 국회 회의록 • 검찰통계시스템
혼합 구조	(유형 4) 클라이언트-서버 조회형	• HTML 문서 내에 인터페이스 정보만 존재하며, 데이터는 서버 통신을 통해 수신됨 • 이용자 조회 요청 이후 서버와의 통신을 통해 인터페이스 및 데이터의 일부를 수신하여 화면에 표시	• 국가통계포털(KOSIS)
	(유형 3) 다중화면 구성형	• 정적 내재형 영역과 클라이언트 서버 조회형 영역이 단일 HTML 문서에 혼재 • 정적 콘텐츠, CSS/JS 기반 동적 표현, 서버 조회 영역 등 다양한 성격의 구성 요소가 혼재하며, 단일한 방식으로 규정되지 않는 복합적인 화면 구조를 가짐 • 대부분의 HTML이 해당 유형에 분포	• e-나라지표 • 기상자료개방포털 • 미디어통계포털 • 서울시 실시간 도시데이터
(유형 1) 동적 렌더링형		• HTML 문서 내 데이터와 인터페이스 정보가 포함되지 않음 • 외부 서버 혹은 애플리케이션 실행에 의해 화면 전체가 동적으로 생성됨 • 페이지 로드 후 특정 프로그램이 실행되면서 외부에서 데이터를 받아 화면 전체를 동적으로 생성	• J대학 학사관리시스템 • 한국은행 경제통계시스템 • 국세청 홈택스

먼저 정적 내재형은 HTML 내에 데이터와 인터페이스 정보가 함께 포함된 구조이다. 이 유형은 페이지 로드 시점에 이미 데이터와 화면 구조가 함께 포함되며 추가적인 서버 요청이나 애플리케이션 실행 없이 전체 내용을 확인할 수 있다. 또한 HTML에 CSS 및 JavaScript 등 외부자원과 연결되는 링크가 존재하므로, 외부자원을 함께 확보한 상태에서 별도의 추가 실행 없이도 화면과 기능을 온전히 재현할 수 있다. 시각화 영역 역시 'SVG', <table> 등 XML 기반 구조로 구성되어 있어 XSLT 변환에 직접 활용 가능한 형태를 가진다.

한편, 일부 사례는 HTML 문서만으로는 화면 전체가 완성되지 않고 서버 요청이나 클라이언트 처리를 통해 화면이 구성되는 특성을 보였다. 이러한 사례들은 HTML 단독으로는 전체 화면을 재현할 수 없고, 데이터 요청 과정이 필수적으로 요구된다는 공통점을 가진다. 이에 본 연구에서는 이러한 구조를 '혼합구조'로 정의하고, 그 내부에서 데이터 요청 방식과 화면 구성 방식의 차이에 따라 '클라이언트-서버 조회형'과 '다중화면 구성형'으로 구분하였다. 클라이언트-서버 조회형은 HTML 문서 내에 입력 폼, 버튼, 레이아웃 등 인터페이스 구조만 존재하고, 실제 데이터는 서버와의 통신을 통해 수신되는 구조이다. 사용자 요청 이후 서버 통신을 통해 콘텐츠를 수신하고 이를 화면에 반영하며, 시각화 영역 역시 서버 응답 결과를 기반으로 구성된다. 이 경우 HTML은 조회 인터페이스 역할만 수행하며, 폼이나 'iframe' 컨테이너와 같은 인터페이스 구조만 제공하므로 전체 HTML 구조는 단순한 형태를 보인다.

반면 다중화면 구성형은 '정적 내재형'과 '클라이언트-서버 조회형'의 요소가 하나의 HTML 문서 내에 혼재된 구조이다. HTML 문서 내에 일부 데이터와 정적 콘텐츠가 포함되어 있으나, 전체 화면은 사용자 요청 이후 서버 응답과 클라이언트 처리를 통해 단계적으로 완성된다. 이 과정에서 CSS와 JavaScript를 기반으로 한 동적 표현과 서버 조회 영역이 함께 작동하며, 다양한 방식의 데이터 요청이 결합되어 사용된다. 또한 실제 웹 기반 행정정보시

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

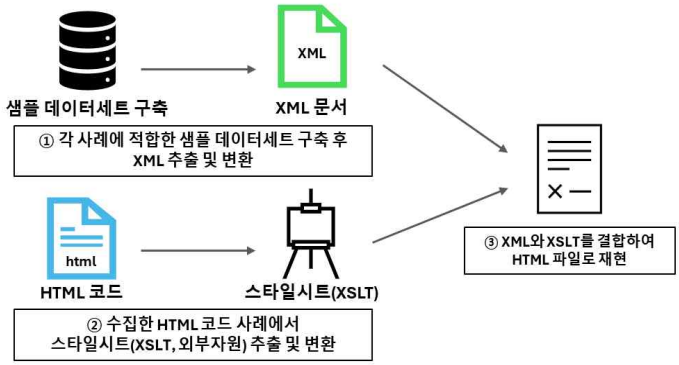
스템의 화면 대부분이 이 유형에 해당하는 것으로 나타났다.

동적 렌더링형은 HTML 문서 내에 데이터와 인터페이스가 포함되지 않으며, 외부 프로그램이나 애플리케이션 실행 이후에야 화면 전체가 동적으로 생성되는 구조이다. 페이지 로드 이후 특정 프로그램이나 프레임워크가 실행되면서 외부 데이터를 수신하고, 이를 기반으로 DOM이 생성·갱신된다. 따라서 HTML 문서만으로는 실질적인 내용을 확인할 수 없으며, 대부분의 콘텐츠가 애플리케이션 형태로 동작한다. 이 유형은 화면 재현을 위해 HTML 외에 실행 환경까지 함께 고려해야 한다는 특징을 가진다.

이와 같이 유형 간 차이는 단순한 구현 방식의 차이를 넘어, XSLT 재현 과정에서 요구되는 처리 방식과 난이도에 직접적인 영향을 미친다. 특히 HTML 내부에 데이터와 화면 구조가 직접 포함된 경우에는 비교적 단순한 변환이 가능하지만, 서버 요청이나 클라이언트 실행에 의존하는 구조는 추가적인 데이터 수집 및 처리 과정이 요구되는 것으로 확인되었다.

4. 유형별 XSLT 재현 방안 적용 및 검증

본 절에서는 앞서 도출한 유형별 특성을 바탕으로 HTML 기반 XSLT 재현 방안을 적용하고, 실제 사용자 인터페이스 재현 가능성을 검증하였다. <그림 8>은 유형별 XSLT 재현 방안의 적용 과정을 나타낸 것이다. 먼저 각 사례의 HTML을 분석하여 화면에 표현되는 데이터 항목과 구조를 확인한 후, 이에 적합한 샘플 데이터세트를 구축하고 XML 형태로 추출·변환하였다. 다음으로 수집한 HTML 사례로부터 XSLT와 외부자원 정보를 추출·변환하였다. 마지막으로 앞서 구축한 XML과 변환 XSLT를 결합하여 HTML 형태의 사용자 인터페이스로 재현하였다.



〈그림 8〉 유형별 XSLT 재현 방안 적용 방법

각 유형의 대표 사례는 해당 유형의 구조적 특성을 가장 명확하게 드러낼 수 있는 사례를 기준으로 선정하였다. 먼저 정적 내재형의 경우 '검찰통계시스템'을 대표 사례로 설정하였다. 이 사례는 'SVG' 기반 차트 데이터가 HTML 내부에 인라인 형태로 포함되어 외부자원에 의존하지 않고도 전체 구조를 파악할 수 있다. 또한 표·차트가 결합된 복합적인 화면 구성을 통해 다양한 재현 요소를 동시에 확인할 수 있어 대표 사례로 채택하였다. 클라이언트-서버 조회형은 '국가통계포털(KOSIS)'을 대표 사례로 선정하였다. 해당 사례는 HTML 문서 내에 실질적인 데이터가 포함되지 않으며, 'form submit'과 'iframe'을 통한 서버 요청 구조가 중심을 이루고, 데이터와 화면이 분리된 전형적인 구조를 보인다는 점에서 해당 유형의 특성을 명확하게 드러낸다. 다중화면 구성형의 대표 사례로는 'e-나라지표'를 설정하였다. 이 사례는 표와 텍스트와 같은 정적 데이터 영역과 'Ajax' 기반 차트와 같은 동적

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

데이터 영역이 단일 HTML 문서 내에 공존하는 구조를 가지므로, 정적 구조와 동적 처리 영역이 결합된 혼합형 특성을 전형적으로 보여주는 사례이다. 동적 렌더링형의 경우 '한국은행 경제통계시스템'을 대표 사례로 선정하였다. 이 사례는 SPA 구조와 'RealGrid', 'RealPivot' 등 외부 프로그램을 기반으로 화면이 구성되며, 이러한 특징은 HTML 내에 렌더링 결과물만 존재한다는 점에서 해당 유형의 XSLT 재현 방안 적용 범위를 드러낼 수 있다고 판단하였다.

XSLT 작성 시 재현 우선순위는 다음과 같이 설정하였다. 첫째, 데이터가 직접 포함된 표 구조를 최우선 재현 대상으로 삼았다. 둘째, 헤더・본문・푸터 등 레이아웃을 구성하는 주요 영역 구분을 재현하였다. 셋째, 탭・버튼 등 UI 컨트롤의 구조적 재현을 시도하였다. 넷째, 'SVG' 및 XML 기반 시각화 요소를 재현 대상에 포함하였다. 반면 동적 처리가 필요한 팝업, 애니메이션 이벤트 등 비핵심 요소는 재현 범위 밖으로 설정하여 핵심 데이터 구조와 표현 요소에 분석을 집중하였다.

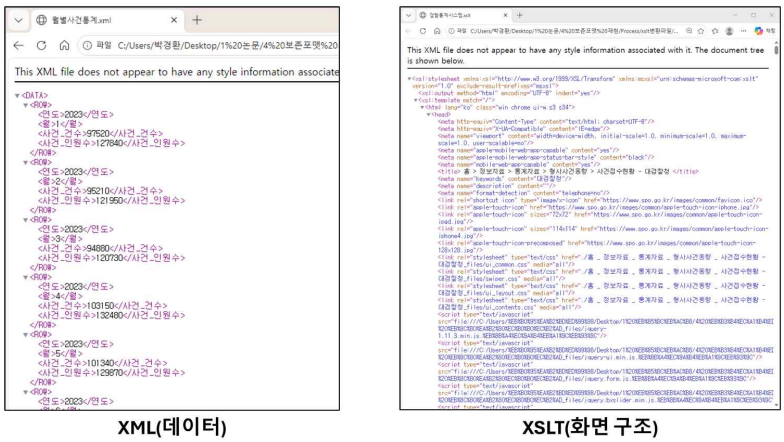
또한 재현 결과의 체계적 비교를 위해 본 연구에서는 <표 8>과 같이 네 가지 평가 기준을 설정하고, 각 유형별 검증 결과를 해당 기준에 따라 분석하였다. 이러한 평가 기준은 HTML 기반 사용자 인터페이스 재현의 수준을 구조, 표현, 데이터 연결, 사용자 경험 측면에서 종합적으로 평가하기 위해 구성하였다.

〈표 8〉 XSLT 재현 방안 적용 결과 평가 기준

평가기준	내용
구조적 재현성	원본 HTML의 DOM 계층 구조와 텍스트 배치가 변환 후에도 논리적으로 유지되는가?
시각 요소 재현성	차트・지도・이미지 등 시각화 요소가 변환 결과에 포함되는가?
데이터 무결성	XML 데이터세트의 요소들이 지정된 인터페이스 위치에 누락 없이 정확히 연결되는가?
시각적 연속성	외부 CSS 및 정적 자원과의 연동을 통해 사용자에게 생산 당시와 유사한 가독성을 제공하는가?

4.1 정적 내재형

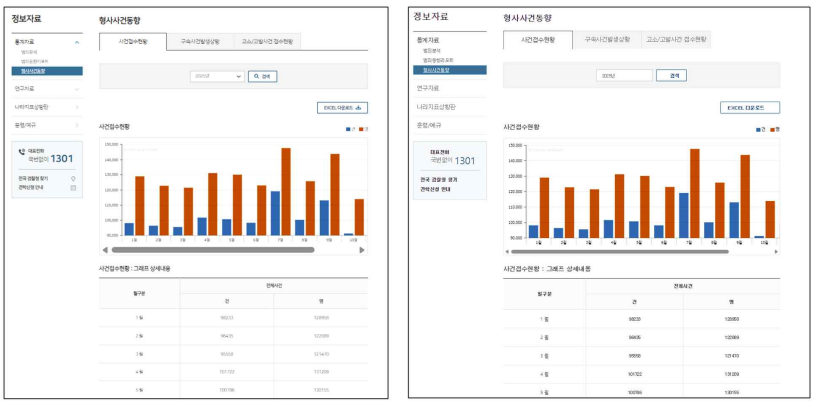
<그림 9>는 '검찰통계시스템'의 사례를 바탕으로 구축한 샘플 XML(데이터)과 XSLT(화면 구조)이다. '검찰통계시스템'은 앞서 분석한 바와 같이 정적 내재형에 해당하는 사례로, HTML 문서 내에 화면 구성에 필요한 데이터와 인터페이스 구조가 대부분 포함되어 있다. 이에 따라 XSLT의 구성 역시 비교적 단순하고 직관적인 형태로 작성될 수 있었다.



XML(데이터) XSLT(화면 구조)

〈그림 9〉 '검찰통계시스템' 산출 XML 및 XSLT

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

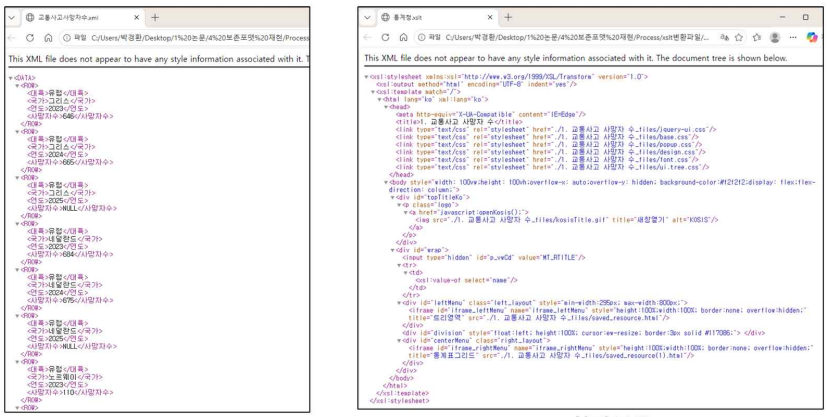


〈그림 10〉 '검찰통계시스템' 재현 결과 비교

<그림 10>은 XSLT 재현 방안을 통해 재현된 '검찰통계시스템 변환 HTML'을 '검찰통계시스템 원본 HTML'과 비교한 것이다. 해당 사례는 정적 내재형의 특성상 데이터와 화면 구조가 HTML 문서 내에 직접 포함되어 있어, XML과 XSLT를 결합한 변환 과정에서도 전체적인 화면 구성이 안정적으로 유지되었다. 재현 결과, 표 구조, 탭 인터페이스, 텍스트 영역 등 주요 화면 요소들은 원본과 대체로 일치하는 형태로 재현되었으며, 외부 경로를 참조하는 일부 이미지 자원의 경우 소실이 확인되었다. 한편 차트 영역의 경우, 해당 데이터가 HTML 내에 인라인 형태로 포함되어 있었던 만큼, 이를 구조화한 도형 데이터를 XML 내에 추가한다면 XSLT 변환 과정에서 차트 영역까지 포함한 보다 완전한 화면 재현이 가능할 것으로 판단된다.

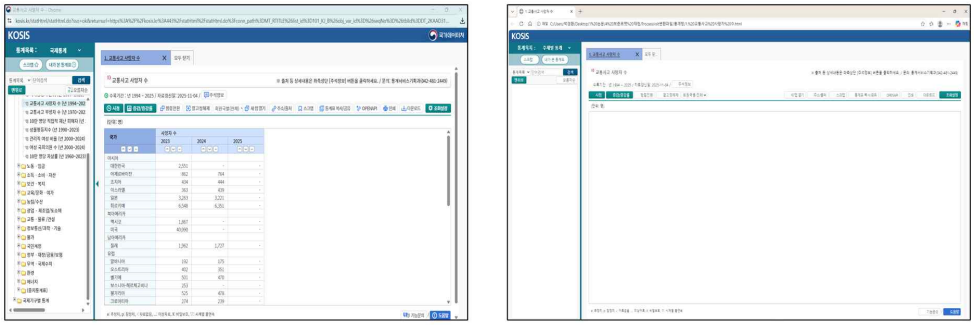
4.2 클라이언트-서버 조회형

<그림 11>은 '국가통계포털(KOSIS)'의 사례를 바탕으로 구축한 샘플 XML(데이터)과 XSLT(화면 구조)이다. 클라이언트-서버 조회형에 해당하는 '국가통계포털(KOSIS)' 사례는 HTML 내에 실질적인 데이터가 포함되지 않고 서버와의 통신을 통해 화면이 구성되는 구조를 가진다. 이에 따라 수집된 HTML에는 서버 요청 및 렌더링 결과만이 존재하였으며, XML와 XSLT는 이러한 렌더링 결과를 기반으로 구성하였다.



〈그림 11〉 '국가통계포털(KOSIS)' 산출 XML 및 XSLT

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131



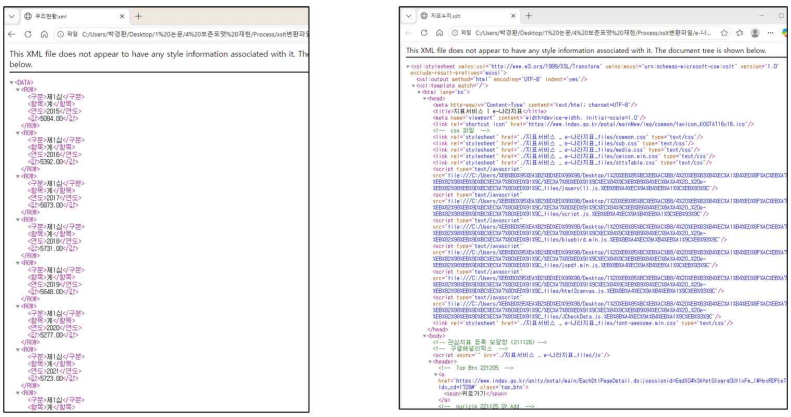
원본 HTML　　변환 HTML(재현 결과)

〈그림 12〉 '국가통계포털(KOSIS)' 재현 결과 비교

<그림 12>는 '국가통계포털(KOSIS) 원본 HTML'과 '국가통계포털(KOSIS) 변환 HTML'을 비교한 것이다. 해당 사례는 서버 응답을 통해 동적으로 구성되는 영역이 화면의 핵심을 이루고 있어, 서버와의 연결 없이는 해당 데이터를 확보하기 어렵다. 이에 서버 통신에 의존하는 요소들을 제거하는 과정에서 좌측 메뉴 영역과 우측의 통계표 영역이 재현 대상에서 제외되었으며, 재현 결과에서도 해당 영역의 내용이 나타나지 않았다.

4.3 다중화면 구성형

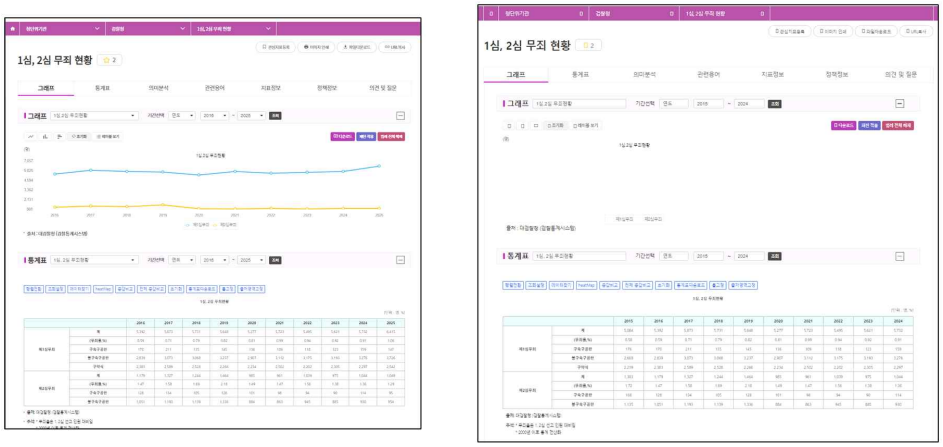
<그림 13>은 'e-나라지표'의 사례를 바탕으로 구축한 샘플 XML(데이터)과 XSLT(화면 구조)이다. 'e-나라지표'는 앞서 분석한 바와 같이 다중화면 구성형에 해당하는 사례로, HTML 문서 내에 일부 정적 데이터와 구조가 포함되어 있으나 차트 영역의 경우 'Ajax'를 통한 서버 요청으로 데이터가 구성되는 혼합적 구조를 가진다. 이에 따라 XML 데이터세트와 XSLT 스타일시트는 HTML 내에 포함된 정적 영역을 중심으로 구성하였다.



XML(데이터)　　XSLT(화면 구조)

〈그림 13〉 'e-나라지표' 산출 XML 및 XSLT

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

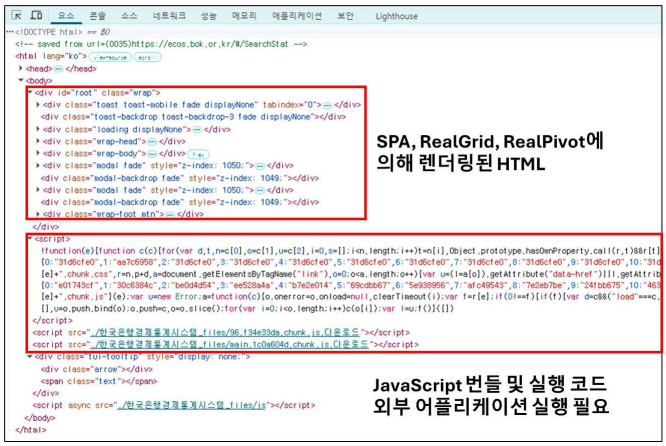


〈그림 14〉 'e-나라지표' 재현 결과 비교

<그림 14>는 'e-나라지표 원본 HTML'과 'e-나라지표 변환 HTML'을 비교한 것이다. 재현 결과, 표 구조와 텍스트 등 HTML 문서 내에 데이터가 직접 포함된 정적 영역은 원본과 대체로 일치하는 형태로 안정적으로 재현되었다. 반면 차트 영역의 경우, 해당 데이터가 서버와의 통신을 통해 동적으로 수신되는 구조로 이루어져 있어 HTML 문서 내에 실질적인 데이터가 존재하지 않았으며, 이로 인해 변환 결과에서 차트 영역의 재현이 불가능하였다.

4.4 동적 렌더링형

<그림 15>는 '한국은행 경제통계시스템'의 원본 HTML 코드를 나타낸 것이다. '한국은행 경제통계시스템'은 앞서 분석한 바와 같이 동적 렌더링형에 해당하는 사례로 HTML 문서 내 실질적인 인터페이스와 데이터가 포함되어 있지 않다. HTML 코드 상단부는 SPA, 'RealGrid', 'RealPivot'과 같은 외부 프로그램에 의해 렌더링된 내용으로 구성되어 있으며, 하단부에는 해당 프로그램의 실행을 위한 JavaScript 번들 및 실행 코드가 위치하고 있다. 즉, 실제 화면은 외부 프로그램이 클라이언트 측에서 실행된 이후에야 동적으로 생성되는 구조로, HTML 내에는 재현에 활용할 수 있는 데이터나 화면 구조가 존재하지 않았다.



〈그림 15〉 '한국은행 경제통계시스템' 재현 대상 HTML 코드

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

앞서 수행한 유형별 적용 결과를 <표 8>의 평가 기준에 따라 종합하면 <표 9>와 같다.

〈표 9〉 XSLT 재현 방안 적용 결과 평가 기준

평가 기준	정적 내재형	클라이언트-서버 조회형	다중화면 구성형	동적 렌더링형
구조 재현성	○	△	△	X
시각 요소 재현성	△	△	○	X
데이터 무결성	○	X	△	X
시각적 연속성	△	X	△	X

(○: 기준 충족 수준이 높음, △: 부분적으로 충족됨, X: 충족되지 않음)

정적 내재형은 주요 DOM 계층 구조와 텍스트 배치가 변환 후에도 유지되며, ‘SVG’ 차트 구조 재현이 가능하나 외부 서버와 연결된 아이콘 이미지 일부가 소실되는 모습을 보였다. 그러나 XML 요소가 지정된 위치에 누락 없이 연결되었으며, 외부 CSS와의 연동을 통해 생산 당시와 유사한 가독성을 제공하는 것으로 나타났다. 이러한 특성은 HTML 자체에 데이터와 구조 정보가 모두 포함되어 있음을 의미하며, XSLT 재현 방안만으로도 구조・데이터・시각 요소를 안정적으로 구현할 수 있는 유형으로 판단된다.

클라이언트-서버 조회형은 서버 응답에 의존하는 영역이 HTML 내에 반영되지 않아 구조 재현이 제한되었으며, 기본적인 틀 구조만 재현 가능한 수준에 그쳤다. 또한 실질적인 데이터가 HTML 내부에 포함되지 않았고, 핵심적인 표 영역이 소실됨에 따라 시각적 연속성 확보에도 어려움이 나타났다. 이러한 한계를 고려할 때 HTML 코드만을 기반으로 한 재현에는 구조적 제약이 존재하며, 실제 렌더링된 화면을 직접 확인하고 이를 참조하여 XSLT를 설계・디자인하는 보완적 접근이 필요하다.

다중화면 구성형은 정적 영역의 구조는 유지되었으나 ‘Ajax’ 기반 동적 영역의 DOM 구조는 재현되지 않았으며, 정적 이미지 요소는 재현 가능하나 ‘Ajax’ 차트는 소실되는 한계를 보였다. 데이터 측면에서는 정적 데이터는 정확히 연결되었으나 동적 데이터 영역은 연결이 불가능하였고, 정적 영역에서는 가독성이 유지된 반면 동적 영역 공백으로 인해 시각적 연속성이 저하되는 것으로 나타났다. 이러한 특성은 정적 영역에 대해서는 XSLT 기반 재현이 유효함을 보여주며, 동적 영역에 대해서는 클라이언트-서버 조회형과 동일하게 렌더링 결과를 참조하는 보완적 설계가 병행될 필요가 있다.

동적 렌더링형은 HTML 문서 전체가 외부 프로그램의 렌더링 결과로 구성되어 있어, 재현에 활용 가능한 DOM 구조 정보, 데이터, 인터페이스 구조를 확인하기 어려웠다. 형식상 DOM 구조는 존재하나, 실제 재현에 활용할 수 있는 데이터나 구조 정보는 포함되어 있지 않았으며, HTML 내에 XML과 연결 가능한 데이터와 인터페이스 요소가 부재하여 데이터 연결이 이루어지지 않았다. 또한 외부 프로그램의 실행 환경 없이는 시각적 출력 자체가 불가능하여, 네 가지 평가 기준 모두에서 XSLT 재현 방안의 적용이 어려운 것으로 확인되었다. 다만 시・도 통합이나 행정체제 개편 등에 따른 대규모 데이터 전환 과정에서 차세대 행정정보시스템을 중심으로 이러한 동적 렌더링형 사용자 인터페이스는 실무에서 지속적으로 확산되고 있다. 이 유형에서 제공되는 화면은 애플리케이션에 의해 실시간으로 생성되는 연속적 데이터로 구성되므로, 정적 문서와 같이 페이지 단위로 분리하여 보존하는 방식으로는 한계가 있다. 특히 지도나 실시간 통계와 같이 동적으로 생성되는 화면은 기록물로서 재현 대상에 해당하지만, 그 결과가 애플리케이션의 실행 과정에 의존한다는 점에서 이관 단계에서 화면 구성 메타데이터(컴포넌트 명세, 필드 정의서, 화면 레이아웃 설계서)를 행정정보 데이터세트와 함께 확보하도록 하는 방안이나 사용자 인터페이스를 생성하는 애플리케이션 자체를 함께 보존하는 방안이 요구된다. 따라서 웹 기반 행정정보시스템의 재현 방식은 텍스트 기반 문서 변환을 넘어, 웹 애플리케이션 전체를 포함하는 총체적 보존 방식으로 확장되어야 한다.

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

5. 결 론

본 연구에서는 행정정보 데이터세트의 화면 재현을 위해 공공기관 행정정보시스템 관련 웹 서비스 10개 사례의 HTML 구조를 분석하고, XSLT 기반 재현 방안의 적용 가능성을 실증적으로 검증하였다. 분석 결과를 토대로 행정정보 데이터세트의 재현 대상 HTML을 정적 내재형, 클라이언트-서버 조회형, 다중화면 구성형, 동적 렌더링형의 네 가지 유형으로 구분하고, 각 유형의 대표 사례에 XSLT를 설계·적용하여 재현 가능 범위와 한계를 확인하였다. 검증 결과, XSLT 기반 재현 방안은 HTML 내에 데이터와 화면 구조가 직접 포함된 정적 내재형에서 가장 효과적으로 적용될 수 있었으며, 서버 요청이나 외부 실행 환경에 대한 의존도가 높아질수록 재현 가능 범위가 축소되고 적용상의 한계가 뚜렷하게 나타났다. 이에 따라 재현 전략 측면에서는 정적 내재형과 다중화면 구성형에는 XSLT 기반 재현을 적용하고, 클라이언트-서버 조회형과 동적 렌더링형에는 렌더링된 화면을 바탕으로 XSLT를 직접 설계하거나, 경량 에뮬레이션 또는 데이터베이스 명세서 기반 보존 방식을 병행하는 혼합 전략을 적용할 수 있다.

본 연구의 의의는 다음과 같다. 첫째, 행정정보 데이터세트의 재현 대상 HTML을 유형화하는 분석 체계를 수립함으로써, 데이터 보존 중심의 기존 기록관리 방식에서 나아가 사용자 인터페이스의 구조와 표현을 함께 보존·재현하기 위한 기술적 접근의 필요성을 제기하였다. 둘째, 재현 과정에서 대상의 특성을 사전에 파악하고 적절한 재현 대상을 선택하는 데 활용 가능한 기초자료를 제안하였다. 셋째, 이론적 수준에 머물렀던 XSLT 재현 방안을 실제 사례에 적용하여 그 가능성과 한계를 실증적으로 제시하였다.

다만 실제 사례 HTML 분석 과정에서 분석 대상이 10개 사례로 한정되어 결과를 일반화하는 데 제약이 따르며, 실제 이관 데이터가 아닌 샘플 데이터세트를 기반으로 검증이 이루어진 만큼 실무 적용 과정에서 추가적인 문제가 발생할 가능성을 배제하기 어렵다. 아울러 XSLT 기반 재현 결과물의 장기보존을 위해서는, 재현 과정에서 생성된 XSLT 파일뿐 아니라 CSS, JavaScript 등 관련 외부 자원 역시 데이터세트와 함께 관리되어야 할 전자기록물로 간주할 필요가 있다. 이에 따라 향후 행정정보 데이터세트 이관 절차에서는 XML, XSLT, CSS 및 기타 외부 자원을 하나의 장기보존패키지로 구성하고, 이들 간의 논리적 연계 관계를 명확히 기술하여 함께 보존·관리하는 방안이 요구된다. 따라서 후속 연구에서는 XSLT 재현 방안의 분석 대상을 확장하고 실제 이관 데이터를 활용한 재현 검증을 수행하는 한편, 실무 적용을 위한 유형별 기술적 가이드라인에 대한 구체화가 이루어져야 할 것이다.

참고문헌

공공기록물 관리에 관한 법률 시행령. 제35506호.

김수영 (2023). 행정정보 데이터세트 이관 방안 적용 사례 연구. 기록과 정보·문화 연구, 16, 7-50. https://doi.org/10.23035/KAICS.2023.1.16.007

변우영, 임진희 (2022). 행정정보 데이터세트 이관도구 SIARD_KR의 개선방안. 정보관리학회지, 39(1), 195-217. https://doi.org/10.3743/KOSIM.2022.39.1.195

양동민, 최광훈, 김지혜, 유남희 (2023). 행정정보 데이터세트의 이관규격의 다양화 및 재현 방안에 관한 연구. 정보관리학회지, 40(4), 167-200. https://doi.org/10.3743/KOSIM.2023.40.4.167

오금용, 황인준 (2002). 유사 패턴을 갖는 HTML 문서의 XML 자동 변환. 정보처리학회논문지 D, 9D(3), 355-364. https://doi.org/10.3745/KIPSTD.2002.9D.3.355

왕호성, 설문원 (2017). 행정정보 데이터세트 기록의 관리방안. 한국기록관리학회지, 17(3), 23-47. https://doi.org/10.14404/JKSARM.2017.17.3.023

유상원, 이형동, 김형주 (2003). 사용자 정보에 기반한 XML문서 전달 시스템. 정보과학회 컴퓨팅의 실제 논문지, 9(5), 487-497.
윤성호, 이정은, 양동민 (2021). 행정정보 데이터세트 보존포맷으로서 SIARD 검증에 관한 연구. 한국기록관리학회지, 21(3), 99-118. http://dx.doi.org/10.14404/JKSARM.2021.21.3.099
이채혁 (2012). 웹 표준에 따르는 HTML, CSS가 웹 디자이너에게 미치는 영향. 석사학위논문, 중앙대학교.
한희정, 윤성호, 오효정, 양동민 (2020). 데이터세트 보존포맷 검증방안에 관한 연구: 재난안전정보 데이터세트의 SIARD 적용을 통해. 정보관리학회지, 37(2), 251-284. https://doi.org/10.3743/KOSIM.2020.37.2.251
Acker, A. (2021). Emulation practices for software preservation in libraries, archives, and museums. Journal of the Association for Information Science and Technology, 72(9), 1148-1160. https://doi.org/10.1002/asi.24482
Chamberland-Thibeault, X. & Hallé, S. (2020). Structural profiling of web sites in the wild. In Web Engineering(ICWE 2020), 27-34. https://doi.org/10.1007/978-3-030-50578-3_3
Clark, C. (2025). Optimize DOM size. Available: https://developer.chrome.com/docs/performance/insights/dom-size?hl=ko
Daszkewicz, J. (n.d.). Difference between static and dynamic web pages. Academia. Available: https://www.academia.edu/23686425/Difference_Between_Static_and_Dynamic_Web_Pages
Goel, A., Zhu, J., Netravali, R., & Madhyastha, H. V. (2022, July 13). Jawa: Web archival in the era of JavaScript. In Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2022), Carlsbad, CA, United States.
Han, K., You, W., Shi, S., Deng, H., & Sun, L. (2024). The doctrine of the mean: Chinese calligraphy with moderate visual complexity elicits high aesthetic preference. International Journal of Human-Computer Interaction, 40(6), 1355-1368. https://doi.org/10.1080/10447318.2022.2144864
Michailidou, E., Harper, S., & Bechhofer, S. (2008, September). Visual complexity and aesthetic perception of web pages. Proceedings of the 26th Annual ACM International Conference on Design of Communication, 215-224. https://doi.org/10.1145/1456536.1456581
Robie, J. (1998, July 20). What is the Document Object Model?. Available: https://www.w3.org/TR/WD-DOM/introduction.html
Vodnik, S. (2011). HTML5 and CSS3, Illustrated Complete. South-Western.
W3C (2014, March 14). The web standards model - HTML, CSS and JavaScript. Available: https://www.w3.org/wiki/The_web_standards_model_-_HTML_CSS_and_JavaScript
W3C (2017, June 8). XSL Transformations (XSLT) Version 3.0. Available: https://www.w3.org/TR/xslt-30/
WHATWG (2026, April 1). HTML5. Available: https://html.spec.whatwg.org/multipage/

• 국문 참고자료의 영어 표기
(English translation / romanization of references originally written in Korean)

Byeon, Woo-Yeong & Yim, Jin Hee (2022). Improvement of administration information dataset transfer tools ‘SIARD_KR’. Journal of the Korean Society for Information Management, 39(1), 195-217. https://doi.org/10.3743/KOSIM.2022.39.1.195
Enforcement Decree of the Public Records Management Act. Presidential Decree No.35506.
Han, Hui-Jeong, Yoon, Sung-Ho, Oh, Hyo-Jung, & Yang, Dongmin (2020). Empirical verification of conversion and restoration of preservation format for dataset: Application of dataset with disaster safety information to SIARD. Journal of the Korean Society for Information Management, 37(2), 251-284. https://doi.org/10.3743/KOSIM.2020.37.2.251
Kim, Su Young (2023). A study on the application of administrative information dataset transfer methods. The Korean Journal of Archival, Information and Cultural Studies, 16, 7-50. https://doi.org/10.23035/KAICS.2023.1.16.007
Lee, Chea-Hyuk (2014). The Influences to Web Designers That Web Standard HTML and CSS Can Make. Master’s thesis, Chung-ang University.

O, Geum-Yong & Hwang, Eenjun (2002). Automatically converting HTML documents with similar pattern into XML documents. The KIPS Transactions:PartD, 9D(3), 355-364. https://doi.org/10.3745/KIPSTD.2002.9D.3.355

Wang, Hosung & Seol, Moon-won (2017). A study on managing dataset records in government information systems. Journal of Korean Society of Archives and Records Management, 17(3), 23-47. https://doi.org/10.14404/JKSARM.2017.17.3.023

Yang, Dongmin, Choi, Kwanghoon, Kim, Ji-Hye, & Yoo, Nam-Hee (2023). Research on diversification of transfer specifications and reproduction methods for administrative information datasets. Journal of the Korean Society for Information Management, 40(4), 167-200. https://doi.org/10.3743/KOSIM.2023.40.4.167

Yoo, Sangwon, Lee, Hyoung-Dong, & Kim, Hyoung Joo (2003). A personalized XML documents delivery system. KIISE Transactions on Computing Practices, 9(5), 487-497.

Yoon, Sung-Ho, Lee, Jung-eun, & Yang, Dongmin (2021). A study on SIARD verification as a preservation format for data set records. Journal of Korean Society of Archives and Records Management, 21(3), 99-118. http://dx.doi.org/10.14404/JKSARM.2021.21.3.099

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

<별첨 1> 행정정보 데이터세트 재현 대상 HTML 상세 분석 결과

분석 기준		J대학 학사관리시스템	국회 회의록	국가통계포털 (KOSIS)	e-나라지표	검찰통계시스템	서울시 실시간 도시데이터	ICT데이터허브 미디어통계포털	한국은행 경제통계시스템	국세청 홈택스	기상자료개방 포털
문법 적합성		O	O	O	O	O	O	O	O	O	O
데이터 구조	DOM 노드 수	3,480개	7,756개	150개	1,227개	1,335개	5,501개	953개	4,322개	7,511개	2,106개
	트리 최대 깊이	35단계	15단계	6단계	18단계	18단계	18단계	15단계	33단계	33단계	18단계
	고유 태그 종류	16종	35종	14종	42종	49종	42종	28종	35종	40종	50종
	링크 수 〈a〉	0개	585개	3개	216개	397개	225개	376개	920개	705개	397개
화면 복잡도	시각 객체 수	161개	160개	112개	87개	7개	314개	33개	93개	125개	55개
	레이아웃 영역 수	8개	6개	4개	5개	6개	11개	6개	5개	181개	7개
	차트 표현 방식	JavaScript 기반	canvas	JavaScript 기반	JavaScript 기반	SVG	canvas	JavaScript 기반 (iframe)	〈table〉	X	JavaScript 기반 (Echarts)
	콘텐츠 유형 구성	이미지, 입력, 텍스트, 탭	텍스트, 목록, 표, 이미지, UI	UI 파라미터, iframe	차트, 표, 텍스트, UI, 링크	차트, 통계표, 텍스트	지도, 차트, 표, 이미지, UI, 텍스트	목록, 링크, 텍스트	목록, 그리드, 버튼	텍스트, 표, UI, 링크, 탭	통계표, 차트, 조회폼, 텍스트, 목록
	지도/차트 외부 라이브러리 사용	O	X	O	O	X	O	X	O	X	O
	기타사항	id 패턴에서 vframe·hframe 분할 구조가 중첩	-	시각적으로 렌더링되지 않는 파라미터 필드가 대부분임	PC·모바일 동일 기능 요소가 이중 마크업 - 중복 제거 처리 필요	-	-	PC·모바일 동일 기능 요소가 이중 마크업 - 중복 제거 처리 필요	-	탭컨트롤 내에 항목별 섹션이 중첩	-
스타일 / 스크립트 의존성	외부 CSS 파일 수	2개	3개	8개	7개	6개	11개	11개	3개	11개	10개
	외부 JS 파일 수	11개	12개	7개	10개	22개	36개	6개	11개	27개	24개
	번들 JS 사용 여부	O	O	O	O	O	O	O	O	O	O
	인라인 CSS 특징 기술	레이아웃이 인라인 스타일 중심으로 구성	레이아웃이 인라인 스타일 중심으로 구성	인라인 CSS는 보조적 수준으로 사용됨	인라인 CSS는 보조적 수준으로 사용됨	인라인 CSS는 보조적 수준으로 사용됨	인라인 CSS는 보조적 수준으로 사용됨	인라인 CSS는 보조적 수준으로 사용됨	인라인 CSS는 보조적 수준으로 사용됨	WebSquare 컴포넌트의 제어에는 인라인 스타일이 병행됨	인라인 CSS는 보조적 수준으로 사용됨
	인라인 JS 특징 기술	초기화, 통신, DOM 처리 함수 포함	-	쿠키처리, 팝업호출, 프레임제어 기능 등 포함	초기화, Ajax, 차트, 다운로드 등 기능 코드 포함	검색, 인쇄, 차트 생성 등의 기능 코드 포함	쿠키처리, 팝업호출, 프레임제어 기능 등 포함	메뉴 이동, 검색, 글자 크기 변경 등의 기능 코드 포함	-	도메인 분기, WebSquare 설정, 초기 부트스트랩 로직이 직접 포함	검색조건 제어, Ajax 조회 등의 코드가 포함
	기타사항	3,422개의 인라인 스타일이 Nexacro 런타임 산출값	JS가 TIFF 원문 로드·발언자 필터 등 부가 기능만 담당	인터페이스의 iframe 분할 구조가 HTML에 존재	-	차트 데이터가 JS 코드 내에 인라인으로 존재함	HTML에는 인터페이스 및 실시간 데이터가 존재하지 않음	-	SPA와 RealGrid, RealPivot 같은 클라이언트 컴포넌트에 의존	-	조회 결과를 서버에서 받아 HTML로 tr·td를 동적 생성하는 구조

https://jksarm.koar.kr
http://dx.doi.org/10.14404/JKSARM.2026.26.2.131

분석 기준		J대학 학사관리시스템	국회 회의록	국가통계포털 (KOSIS)	e-나라지표	검찰통계시스템	서울시 실시간 도시데이터	ICT데이터허브 미디어통계포털	한국은행 경제통계시스템	국세청 홈택스	기상자료개방 포털
동적 기능 수준	이벤트 처리 구조	페이지 종료 시 로그아웃 수행	함수 호출을 통해 이벤트 처리	마우스 드래그 이벤트와 jQuery 기반 이벤트 처리	jQuery 및 인라인 이벤트 속성으로 처리	jQuery 및 인라인 이벤트 속성으로 처리	onclick, 함수 호출을 통해 이벤트 처리	jQuery 및 onclick으로 처리	번들 JS 실행을 통해 처리	WebSquare 내부 함수 호출을 통해 처리	jQuery 및 onclick으로 처리
	DOM 조작 코드	동적 요소 생성 삽입 삭제 코드 포함	-	CSS와 DOM 이벤트 제어를 통해 레이아웃 크기 변경 및 iframe 콘텐츠 로딩	jQuery 등을 이용해 DOM 조작	jQuery 등을 이용해 페이지 체류과 만족도 점계 값 갱신	jQuery 등을 이용해 DOM의 상태를 계속 변경함	jQuery 등을 이용해 DOM 조작	클라이언트 측 DOM 구성 · 갱신을 통해 조작됨	jQuery 등을 이용해 DOM 조작	jQuery 등을 이용해 DOM 조작
	데이터 요청 구조	jQuery 기반 비동기 요청 구조	Ajax로 이미지 데이터(TIFF) 비동기 요청	form submit, iframe를 통해 서버로부터 데이터 요청	Ajax, iframe src 변경을 통해 서버 데이터를 반복적으로 요청	Ajax로 이미지 데이터(TIFF) 비동기 요청	외부 JS 모듈과 지도 타일 호출을 통해 서버로부터 실시간 데이터 갱신	Axax, iframe, form submit로 데이터를 요청	webpack chunk 분할 로딩 코드 등을 통해 데이터를 동적으로 요청함	WebSquare, iframe으로 데이터를 요청	Ajax, form submit로 데이터를 요청
	비HTML 기반 요소	Nexacro, iframe 포함	-	iframe 포함	canvas, iframe 포함	〈svg〉 기반 amCharts 포함	canvas 포함	Ajax, iframe 포함	realgrid, treegrid, realpivot, SPA 등 포함	WebSquare, iframe 포함	Bcharts, Ajax 포함
	기타사항	-	서버 측에서 완성된 HTML을 제공	iframe 내부 파일 분석 필요	페이지 로드 시 의견 · 통계표 조회 데이터를 호출하는 Ajax 프로그램	서버 측에서 완성된 HTML을 제공	스냅샷 당시의 실시간 데이터 수치를 포함한 팝업 컨테이너와 UI 레이블 텍스트가 HTML에 존재함	HTML에 포털형 인터페이스와 통계표 탐색 구조 구현	-	WebSquare 컴포넌트 렌더링 결과가 HTML 문서 내에 포함됨	HTML에 인터페이스 구조와 검색 조건 폼 구현

http://dx.doi.org/10.14404/JKSARM.2026.26.2.131